National Semiconductor™

ADVANCED INFORMATION
May 1997

Support: ns486@arador.nsc.com

NS486™SXL Optimized 32-bit 486-class Controller With On-chip Peripherals for Embedded Systems

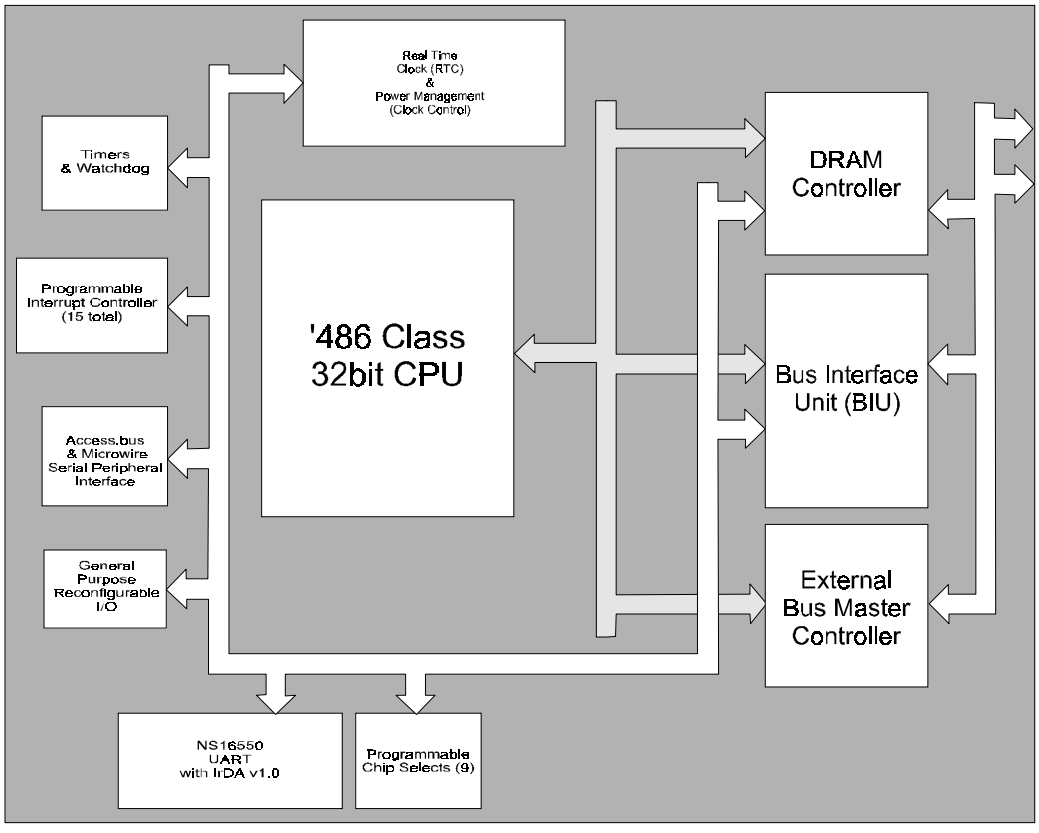
General Description

The NS486SXL is a highly integrated embedded system controller incorporating an Intel486™-class 32-bit processor along with all of the necessary System Service Elements, implementing a true "system on a chip." It is ideally suited for a wide variety of applications running in a segmented protect-mode environment. The NS486SXL is the second member of the NS486 family.

Key Features

- 100% compatible with VxWorks®, VRTX®, QNX® Neutrino, pSOS+™, and other popular real-time executives and operating system kernels
- Intel486™ instruction set compatible (protected mode only) with optimized performance
- Operation at 25 MHz with 5 Volt supply
- Low cost 132-pin PQFP package
- Industry standard interrupt controller, timers, and real time clock
- Protected WATCHDOG™ timer
- Optimized DRAM Controller (supports two banks, up to 8 Mbytes each)
- Up to nine versatile, programmable chip selects
- Up to eight external interrupts directly supported, and additional interrupt expansion through an external PIC interface
- Glueless interface to ISA-type peripherals
- Arbitration support for auxiliary processor
- Support for External Bus Masters, allowing them to access DRAM and on-chip Peripherals
- MICROWIRE™/Access.bus synchronous serial interfaces
- UART with IrDA v1.0 (Infrared Data Association) port
- Reconfigurable I/O: Up to 28 I/O pins can be used as general purpose bidirectional I/O lines
- Flexible, programmable, multilevel power saving modes maximize power savings
- Programming model compatible with the NS486SXF where possible

NS486SXL Single-Chip Embedded Controller



NS486™SXL Optimized 32-bit 486-class Controller With On-chip System Service Elements for Embedded Systems

Contents

1.0 SXL Pin Description Tables 7

2.0 System Overview 17

2.1 NS486SXL System Overview 17

2.2 32-bit Processor Core 18

2.3 System Service Elements 18

2.3.1 DRAM Controller 19

2.3.2 Programmable Interval Timer 19

2.3.3 WATCHDOG Timer 19

2.3.4 Interrupt Controller 19

2.3.5 Real Time Clock/Calendar 19

2.3.6 Power Management Features 19

2.4 NS486SXL System Bus 20

2.5 Other On-board Peripherals 21

2.5.1 Reconfigurable I/O Lines 21

2.5.2 MICROWIRE/Access.bus Interface . . 21

2.5.3 UART Serial Port 21

2.6 ICE Support 22

2.7 Other Issues 22

3.0 Programming the NS486SXL 23

3.1 Overview 23

3.2 I/O Address Map 24

3.3 System Service Elements 28

3.3.1 The DRAM Controller 28

Reset Condition 29

DRAM Parity Checking 29

Refresh 29

RAS Recovery (pre-charge) Rate 29

The DRAM Registers 30

3.3.2 The Programmable Interval Timer (PIT) 39

Programming the PIT System 40

Control Word Register 40

Counter Write Operations 41

Counter Read Operations 41

Counter Latch Command 42

Read-Back Command 42

External Timer Control Signals 43

Timer I/O Control Register 44

Timer Clock Register 45

Mode Descriptions 45

Gate Input 47

3.3.3 The WATCHDOG Timer 48

WATCHDOG Timer Control Register . . . 48

Retriggering the WATCHDOG Timer Count to Prevent Resets 49

Interrupt Request Supported 49

Programming Timer 2 49

3.3.4 The MICROWIRE or Access.bus Interface 50

MICROWIRE Interface 50

MICROWIRE Transfers 51

The Access.Bus Interface 53

Access.bus Data Transfers 54

Addressing Transfer Formats 56

MICROWIRE/Access.bus Control Register (MACON) 57

MICROWIRE Registers 58

MICROWIRE/Access.bus Interface Programming Notes 62

4.0 The System Bus 63

4.1 Bus Interface Unit (BIU) 63

4.1.1 Internal Peripheral Accesses 64

NS486SXL Peripheral Control Registers 64

Global Enable 65

Accessing The Internal UART 65

Accessing Other Internal Peripherals 65

4.1.2 Interrupt Acknowledge Cycles 65

4.1.3 HALT Cycles 65

4.1.4 SHUTDOWN Cycles 65

4.2 External Bus Cycles 66

4.2.1 ISA-like Bus Operation 66

4.2.2 ISA-like Bus Timing 67

4.3 Device Configuration at RESET 69

4.3.1 Boot-up ROM BIOS Size 69

4.3.2 Normal Mode vs. ICE Mode 69

4.4 Chip Select Logic 70

4.4.1 Boot ROM Data Bus Size 70

4.4.2 Other Chip Selects' Data Bus Size . . . 70

4.4.3 Programmable Chip Select Architecture 70

4.4.4 Programming Logical Chip Selects . . 72

4.4.5 Combining Chip Selects 74

4.4.6 Programmable Access Timing 74

4.5 Bus Interface Unit Registers 79

4.5.1 Bus Interface Unit Control Register 1 79

4.5.2 Bus Interface Unit Control Register 2 80

4.5.3 Chip Select Base Address Registers 80

Chip Select Base Address Register 1 . . . 80

Chip Select Base Address Register 2 . . . 81

Chip Select Base Address Register 3 . . . 81

Chip Select Base Address Register 4 . . . 81

Chip Select Base Address Register 5 . . . 81

Chip Select Base Address Register 6 . . . 81

Chip Select Base Address Register 7 . . . 81

Chip Select Base Address Register 8 . . . 81

4.5.4 Chip Select Address Mask Registers 81

Chip Select Address Mask Register 1 . . 82

Chip Select Address Mask Register 2 . . 82

Chip Select Address Mask Register 3 . . 82

Chip Select Address Mask Register 4 . . 82

Chip Select Address Mask Register 5 . . 82
Chip Select Address Mask Register 6 . . 82
Chip Select Address Mask Register 7 . . 82
Chip Select Address Mask Register 8 . . 82
4.5.5 Chip Select Enable Register 82
4.5.6 Chip Select Type Register 83
4.5.7 Chip Select Access Time Registers 83
Chip Select Access Time Register 1 . . . 83
Chip Select Access Time Register 2 . . . 83
Chip Select Access Time Register 3 . . . 84
Chip Select Access Time Register 4 . . . 84
Boot ROM Access Time Register 85
4.5.8 Cacheable Chip Selects Register 85
4.5.9 External Chip Selection Registers 86
Chip Select Local Device Register. 86
Boot ROM Selection Register 86
External Chip Select Selection Register 1. 86
External Chip Select Selection Register 2. 87
External Chip Select Selection Register 3. 87
External Chip Select Selection Register 4. 87
External Chip Select Selection Register 5. 87
External Chip Select Selection Register 6. 88
External Chip Select Selection Register 7. 88
External Chip Select Selection Register 8. 88
4.5.10 Programmable 16-bit Logical Chip Select Register 89
4.6 Auxiliary Processor, Shared Memory Interface 89
4.6.1 Auxiliary Processor Communications. 89
4.6.2 CPU Controlled Transfers 89
4.6.3 Auxiliary Processor Chip Select Selection Register 92
4.7 Device ID and Revision Registers 92
4.7.1 Device ID Register 92
4.7.2 Device Revision Register 92

5.0 NS486SXL Bus Master Operation 93

5.1 INTRODUCTION 93
5.2 Access Sequence 94
5.2.1 Notes on HOLD Request 101
5.2.2 Notes on RDY 101
5.2.3 Notes on CS16 101
5.2.4 Notes on BDIR 101
5.3 Performance Considerations 102

6.0 Reconfigurable I/O (RIO) 107

6.1 Reconfigurable I/O Functional Description 107
6.1.1 Reconfigurable I/O Control Register 107
6.1.2 Data Direction Register 108
Data Direction Register Definitions 109
Data Port Out Register. 116
Data Port In Register 116

7.0 Device Specifications 117

7.1 DC Electrical Specifications 5V ± 5% 117
7.1.1 Recommended Operating Conditions 117
7.1.2 Absolute Maximum Ratings (Notes 2 and 3) 117
7.1.3 Capacitance: TA = 25°C, f = 1 MHz . 117
7.1.4 DC Characteristics 118
External Bus 118
DRAM Control Unit. 118
Auxiliary Processor Interface 118
IrDA Infra Red/UART. 119
External Bus Control 119
Oscillator (CPUX1/CLK) 119
Real Time Clock (RTCX1/CLK) 119
Timer 119
General Purpose Chip Selects 120
Interrupt Controller 120
3-Wire I/O (& Access.bus) 120
7.2 General AC Specifications 121
7.2.1 Power Ramp Times. 123
7.2.2 PWRGOOD and Power Rampdown Timing 123
7.3 AC Switching Specifications 124
7.3.1 DRAM Interface Timing Specification. . 125
7.3.2 ISA-like Bus Cycles Timing Specification 127
7.3.3 Ready Feedback Timing Specifications . 129
7.3.4 OSCX1 AC Specification 129
PIC AC Specs 130
MICROWIRE (3-Wire) & Access.bus . . 131
FIFO UART. 132
Appendix A: External Interrupt Controller . . . 135
Appendix B: Dealing with Unused Signals on the NS486SXL 137

SXL Pin Description Tables

NS486SXL Package Pinout Diagram. 7
Bus Interface Unit Pins. 8
External Bus Master Interface Pins 10
DRAM Control Pins. 10
Power Pins 11
Reset Logic Pins. 11
General Purpose Chip Select Pins 11
Auxiliary Processor Interface Pins. 11
Test Pins 12
Interrupt Control Pins. 12
Real Time Clock Pins. 12
Oscillator Pins 12
16550 UART Pins 13
Timer Pins 14
3-Wire Serial I/O Pins 14
Summary of Reconfigurable I/O Pins 15

System Overview

NS486SXL System Overview 17
NS486SXL Internal Resource to Pins Map 17
32-bit Processor Core 18
System Service Elements 18
NS486SXL System Bus 20
NS486SXL Internal Busses 20
Other On-board Peripherals 21
ICE Support 22
Other Issues 22

Programming the NS486SXL

Overview 23
NS486SXL I/O Utilization. 23
I/O Address Map 24
I/O Address Map 24
System Service Elements 28
Memory Addressing. 28
Valid DRAM Address Chip Select Generation 33
DRAM Read Cycle (4 cycle page miss) 36
DRAM Write Cycle (4 cycle page miss) 36
DRAM Read Followed by DRAM Write (4 cycle page miss). 36
CAS Before RAS Refresh Cycle (4 cycle page miss) 37
DRAM Read Cycle (3 cycle page miss) 37
DRAM Write Cycle (3 cycle page miss) 37
DRAM Read Followed by DRAM Write (3 cycle page miss). 38
CAS Before RAS Refresh Cycle 38
Programmable Interval Timer 39
A MICROWIRE System Example. 50
MICROWIRE Normal SCLK Mode Timing Diagram 51
MICROWIRE Alternate SCLK Mode Timing Diagram 52
A Sample Access.bus System 53

Multiple Resources . . . 53
Bit Transfer . . . 54
Start and Stop Conditions . . . 54
Access.bus Data Transfer . . . 55
Access.bus Acknowledge Cycle . . . 55
A Complete Access.bus Data Transfer . . . 56

The System Bus

Bus Interface Unit (BIU) . . . 63
NS486SXL Internal Buses . . . 63
External Bus Cycles . . . 66
ISA-like Bus Cycle Timing . . . 68
Device Configuration at RESET . . . 69
Chip Select Logic . . . 70
Logical Chip Selection . . . 71
External Chip Selection . . . 71
Address Masking . . . 72
No Wait States, No Command Delays, External Interface Bus Cycles (Internal access timing is similar) . . . 75
No Wait States, One Command Delay, External Bus Cycles. . . . 75
One Wait State, No Command Delays, External Interface Bus Cycles . . . 75
One Wait State, One Command Delay, External Interface Bus Cycles . . . 76
Two Wait States, One Command Delay, External Interface Bus Cycles . . . 76
One Command Delay, Ext. Device Inserts Wait State, External Interface Bus Cycles . . . 76
One Command Delay, Ext. Device Inserts Wait State, One Clock Period of RDY Extension, External Interface Bus Cycles . . . 77
One Command Delay, Ext. Device Inserts Wait State, Two Clock Periods RDY Extension, External Interface Bus Cycles . . . 77
16-bit to 8-bit Cycle Translations . . . 78
Bus Interface Unit Registers . . . 79
Auxiliary Processor, Shared Memory Interface . . . 89
Auxiliary Processor Shared Memory Block Diagram . . . 91
Shared Memory Single Access (CPU Controlled) . . . 91
Device ID and Revision Registers . . . 92

NS486SXL Bus Master Operation

INTRODUCTION . . . 93
Bus Mastering System Block Diagram. . . . 93
Access Sequence. . . . 94
HOLD and HLDA relationship . . . 94
MAE Example . . . 95
External Master DRAM Access . . . 96
Non-DRAM External Master Access . . . 96
External Master DRAM Write . . . 97
External Master Non-DRAM Write . . . 98
RDY During DRAM Write Access . . . 99
RDY During non-DRAM Write Access. . . . 99

RDY During DRAM Read 100
RDY During non-DRAM Read 101
Performance Considerations. 102
Shortest Possible System Bus Access 103
DRAM Page Hit Write Access. 103
DRAM Page Hit Read Access 104
Read Access to Internal Peripheral 104
Fast Back-to-Back Accesses 105

Reconfigurable I/O (RIO)

Reconfigurable I/O Functional Description 107
Reconfigurable I/O Control Register 107
RIO Pins and Associated Register Bits 108
Reconfigurable I/O Data Direction Register (bits 31-24) 109
Reconfigurable I/O Data Direction Register (bits 23-16) 111
Reconfigurable I/O Data Direction Register (bits 15-8) 113
Reconfigurable I/O Data Direction Register (bits 7-0) 114

Device Specifications

DC Electrical Specifications 5V ± 5% 117
General AC Specifications 121
Switching Characteristic Measurement Waveforms 121
More Switching Specifications 122
Power Supply Rise and Fall 123
Vdd Rise and Fall Times 123
Vdd Rampdown vs. PWRGOOD. 123
PWGOOD in relation to Vdd 123
AC Switching Specifications 124
DRAM Timing Diagram 125
4 Cycle Page Miss Preliminary Specifications. 125
3 Cycle Miss Preliminary Specifications 126
ISA-like Bus Timing Diagram 127
No Command Delay ISA-like Bus Specifications 127
One Programmed Command Delay ISA-like Bus Specifications 128
Ready Feedback Timing Diagram 129
Ready Signal Timing Specifications 129
TTL Clock Input Timing Diagram 129
TTL Clock Input Specification. 129
PIC Timing Diagram 130
PIC Timing Specifications 130
Access.bus Timing Diagram 131
Access.Bus Timing Specifications 131
UART Baud Rate and Infrared Clocks. 132
UART MODEM Control Timing. 133

1.0 SXL Pin Description Tables

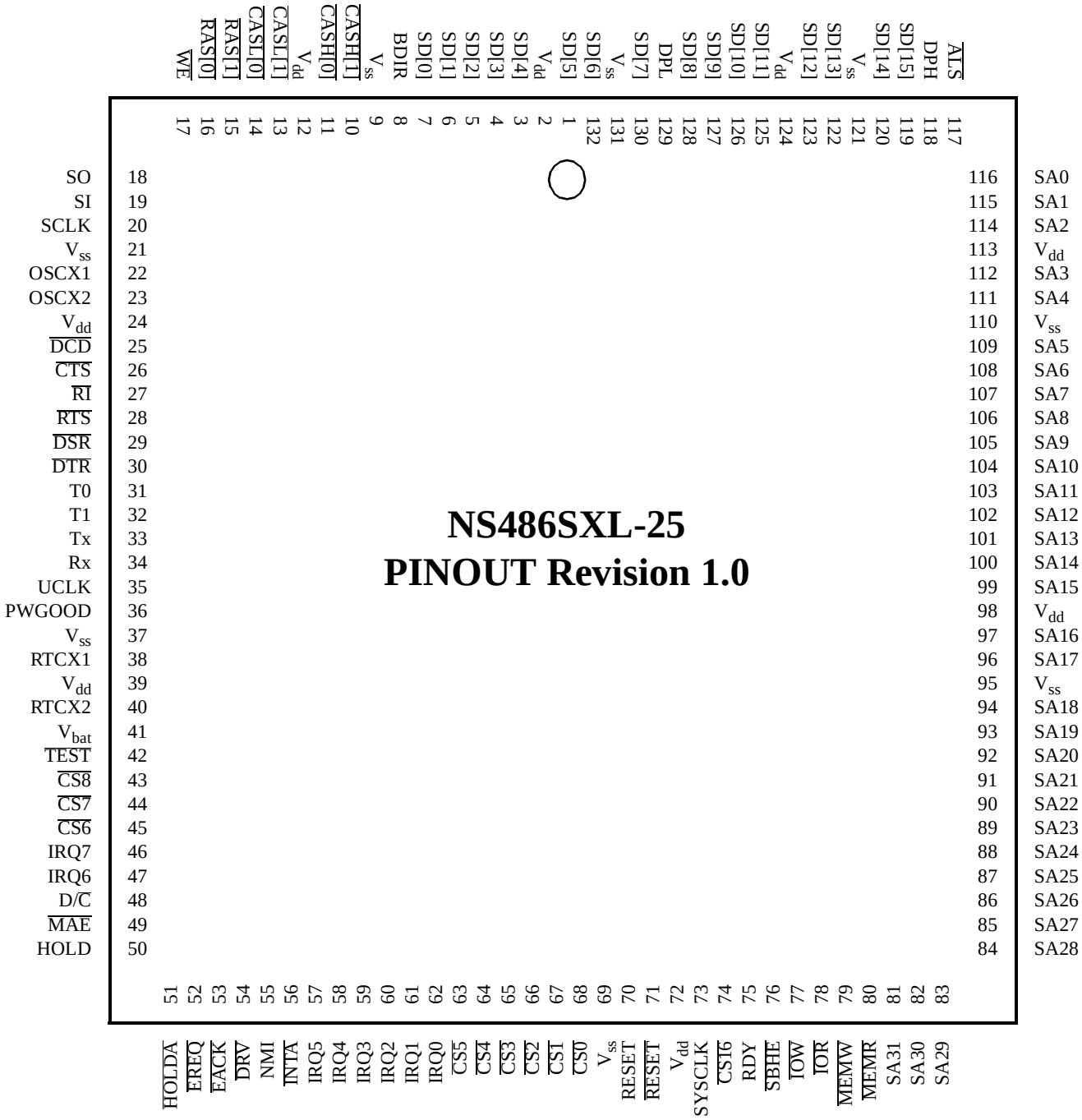


Figure 1-1 NS486SXL Package Pinout Diagram

Note: In the above figure and in the following tables, all active low signals are shown with an overbar.

Table 1-1: Bus Interface Unit Pins

Symbol	Pins	Type	Function
SA[31:0]	81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 96, 97, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 111, 112, 114, 115, 116	I/O	**S**ystem **A**ddress bus. These input-output signals carry the latched address for the current access. DRAM accesses multiplex the row and column addresses for the DRAMs on the SA[12:1] pins. **Note: An incompatibility was introduced into the first silicon of the 'SXL. During Interrupt Acknowledge cycles, the internal master interrupt controller's cascade line signals, CAS[2:0], are driven onto SA[31:29], respectively. Formerly the CAS[2:0] signals were driven onto SA[25:23] in the 'SXF.** The SA[31:0] pins are inputs when an External Master is in control of the bus, except when the 'SXL does a DRAM access for the External Master (see $\overline{\text{MAE}}$, below).
SD[15:0]	119, 120, 122, 123, 125, 126, 127, 128, 130, 132, 1, 3, 4, 5, 6, 7	I/O	**S**ystem **D**ata bus: This bi-directional data bus provides the data path for all memory and I/O accesses. During transfers with 8-bit devices, the upper data byte is not used (SD[15:8]).
$\overline{\text{ALS}}$	117	O	**A**ddress **L**atch **S**trobe. This pulse is produced by a variety of bus related activities. The ALS strobe will go low every time a bus cycle is initiated by the internal CPU, even if the cycle is killed due to an internal instruction-cache "hit." The strobe will also go active for each DRAM access, and each eight-bit access for 16-to-8 bit translations by the Bus Interface Unit (BIU). The strobe will be produced for internal and external I/O accesses as well. Finally, the strobe will go active low during External Bus Master accesses so the BIU can indicate to the internal CPU that it should "snoop" an access to possibly invalidate cache entries.
$\overline{\text{SBHE}}$	76	I/O	**B**yte **H**igh **E**nable. This active-low signal is driven when the address is asserted by the CPU. External 16-bit devices should use this signal to help them determine that a data byte is to be transferred on the upper byte of the System Data bus (SD[15:8]). Eight-bit devices should ignore this signal. The 'SXL bus interface will automatically translate 16-bit requests from the internal CPU into two eight bit accesses for external memories and peripherals that do not assert $\overline{\text{CS16}}$. This pin becomes an input when an External Master is in control of the bus. An External Master should drive $\overline{\text{SBHE}}$ appropriately according to the type of access it is requesting, and be prepared to handle 8-bit devices if a 16-bit access is attempted and no $\overline{\text{CS16}}$ is produced. The 'SXL bus interface will automatically translate 16-bit accesses from the External Master into two eight-bit accesses for internal peripherals. The 'SXL will also respond with $\overline{\text{CS16}}$ on accesses to internal peripherals and accesses to any programmed Chip Select that has the "force 16-bit" feature enabled. **$\overline{\text{SBHE}}$ Truth Table** SBHE SA[0] Function 0 0 The bus master is requesting a 16-bit transfer 1 0 An 8-bit transfer on the low-byte is requested 0 1 An 8-bit transfer on the high-byte is requested 1 1 Illegal case

D/$\overline{C}$	48	O	Data/$\overline{Control}$. This output is provided to indicate what kind of access the 'SXL internal CPU is making. During the time that an External Master controls the bus, D/$\overline{C}$ will be high, indicating Data accesses. D/$\overline{C}$ is high for I/O and Memory accesses that are considered "data" by executing instructions. D/$\overline{C}$ is low for code fetches from memory, interrupt acknowledge cycles and Halt/Special bus events.
BDIR	8	O	**B**uffer **DIR**ection. This output is provided to reduce external logic if an external data-bus buffer is required in the user's design. The BDIR signal is high whenever the buffer should be driving from the 'SXL pins out to the buffered ISA-like bus. BDIR also works correctly if an External Master is designed into the system, however, the External Master must always be on the buffered side of the bus in this case. BDIR will only go low during reads from the buffered bus, or accesses to internal peripherals or DRAM by an External Master.
$\overline{IOR}$	78	I/O	**IO R**ead command. This active-low signal instructs an I/O device to place data onto the system data bus. An input when an External Master controls the bus.
$\overline{IOW}$	77	I/O	**IO W**rite command. This active-low signal indicates to an I/O device that a write operation is in process on the system bus. An input when an External Master controls the bus.
$\overline{MEMR}$	80	I/O	**MEM**ory **R**ead command. This active-low signal instructs a memory mapped device to place data onto the system data bus. An input when an External Master controls the bus.
$\overline{MEMW}$	79	I/O	**MEM**ory **W**rite command. This active-low signal indicates to a memory mapped device that a write operation is in process on the system bus. An input when an External Master controls the bus.
$\overline{CS16}$	74	I/O	**C**hip **S**elect **16**-bit. This active-low feedback signal indicates that the device being accessed is a 16-bit device. This signal should be pulled-up and driven by external devices with an open collector driver. If a chip select is programmed to force 16-bit accesses, this signal will be asserted (low) during the access. When an External Master controls the bus, the 'SXL will also drive this signal low for accesses to internal peripherals or DRAM.
RDY	75	I/O	**R**ea**DY**. An external device may drive this signal inactive low to insert wait states and extend the external bus cycle. This signal should be pulled-up and driven with an open collector or be TRI-STATE driven. When an External Master controls the bus, it must honor the $\overline{RDY}$ signal as the 'SXL will drive this signal low as appropriate for accesses to internal peripherals or DRAM and bus snooping.

Table 1-2: External Bus Master Interface Pins

Symbol	Pins	Type	Function
HOLD	50	I	**HOLD** Request from External Master. The external master will assert this signal high in order to request the bus from the 'SXL CPU. The external master can hold the bus indefinitely, so care should be taken to ensure that the HOLD is released in time for the CPU to service any real-time requirements (e.g. Interrupts, etc.).
$\overline{\text{HLDA}}$	51	O	**HoLD A**cknowledge from 'SXL. When the 'SXL CPU grants the bus to an external master, then this signal is asserted (low). Once $\overline{\text{HLDA}}$ is asserted, the external master is responsible for driving the address and control signals ($\overline{\text{MEMR}}$, $\overline{\text{MEMW}}$, $\overline{\text{IOR}}$, $\overline{\text{IOW}}$, $\overline{\text{SBHE}}$) on the bus. If there are bi-directional buffers on the address and control lines, then $\overline{\text{HLDA}}$ should be used to set the direction of the buffers.
$\overline{\text{MAE}}$	49	O	**M**aster **A**ddress **E**nable. During $\overline{\text{HLDA}}$, if the 'SXL requires that the External Master tri-state its addresses (e.g. to complete a DRAM access) then $\overline{\text{MAE}}$ will be de-asserted (high). $\overline{\text{MAE}}$ should be used to control the External Master's tri-state address lines or for the enable of the bi-directional address bus buffer chips. $\overline{\text{MAE}}$ will normally be asserted (low).

Table 1-3: DRAM Control Pins

Symbol	Pins	Type	Function
$\overline{\text{RAS}}$[1:0]	15, 16	O	**R**ow **A**ddress **S**trobe. On the falling edge of these active-low signals, Bank 1 and Bank 0 respectively, should latch in the row address off of SA[12:1]. If only one bank of DRAM is supported, $\overline{\text{RAS0}}$ will support that bank and $\overline{\text{RAS1}}$ will be unused.
$\overline{\text{CASH}}$[1:0]	10, 11	O	**C**olumn **A**ddress **S**trobe (**H**igh Byte). These active-low signals indicate when the column access is being made to the high byte of DRAM Bank 1 and DRAM Bank 0 respectively. If only one bank of DRAM is supported, $\overline{\text{CASH0}}$ will support the high byte of that bank and $\overline{\text{CASH1}}$ will be unused.
$\overline{\text{CASL}}$[1:0]	13, 14	O	**C**olumn **A**ddress **S**trobe (**L**ow Byte). These active-low signals indicate when the column access is being made to the low byte of DRAM Bank 1 and DRAM Bank 0, respectively. If only one bank of DRAM is supported, $\overline{\text{CASL0}}$ will support the low byte of that bank and $\overline{\text{CASL1}}$ will be unused.
$\overline{\text{WE}}$	17	O	**W**rite **E**nable. Active low signal for write operations on DRAM.
DPH, DPL	118, 129	I/O	DRAM **D**ata **P**arity. DRAM data parity may be enabled or disabled; if disabled these two pins will be unused. Otherwise, for DRAM writes the SXL's DRAM Controller will generate odd parity and drive the odd parity onto these two pins. For DRAM reads the SXL's DRAM Controller will read the values driven on these two pins and check it for odd parity in association with the appropriate data byte.

Table 1-4: Power Pins

Symbol	Pins	Type	Function
V_{DD}	2, 12, 24, 39, 72, 98, 113, 124	Power	+5V power to core and I/O.
V_{SS}	9, 21, 37, 69, 95, 110, 121, 131	Ground	Ground to core and I/O.

Table 1-5: Reset Logic Pins

Symbol	Pins	Type	Function
RESET	70	O	**RESET** system output driver: This active high signal resets or initializes system peripheral logic during power up (PWGOOD) or due to a WATCHDOG Reset.
$\overline{\text{RESET}}$	71	O	Inverse of **RESET** for peripherals requiring active low reset.
PWGOOD	36	I	**P**o**W**er **GOOD**. This active-high (schmitt trigger) input will cause a hardware reset to the **NS486SXL** whenever this input goes low. This pin will typically be driven by the power supply and PWGOOD will remain low until the power supply determines that stable and valid voltage levels have been achieved.

Table 1-6: General Purpose Chip Select Pins

Symbol	Pins	Type	Function
$\overline{\text{CS}}$[0]	68	O	**C**hip **S**elect **0**: This output is used as the chip-select for the system boot ROM. It defaults to the upper 64k Bytes of memory.
$\overline{\text{CS}}$[8:1]	43, 44, 45, 63, 64, 65, 66, 67	O	**C**hip **S**elect **1 to 8**. These pins can be programmed to be either memory or I/O mapped chip selects, which are used for glueless connection to external peripherals.

Table 1-7: Auxiliary Processor Interface Pins

Symbol	Pins	Type	Function
$\overline{\text{EREQ}}$	52	O	**E**xternal bus **REQ**uest (active-low) to an auxiliary processor. This signal is asserted whenever the auxiliary processor feature of a programmable chip select is enabled. This is used to request access to a shared memory from another processor.
$\overline{\text{EACK}}$	53	I	**E**xternal bus **ACK**nowledge (active-low) from an auxiliary processor.
$\overline{\text{DRV}}$	54	O	Auxiliary processor shared memory **DRi**V**e** control signal. Once access is granted to the shared memory, this signal is asserted to enable the address, data and control signal buffers to drive the shared memory pins.

Table 1-8: Test Pins

Symbol	Pins	Type	Function
$\overline{\text{TEST}}$	42	I	Reserved for testing and development system support. Normally pulled high. A small number of test modes are documented for use by the customer. While $\overline{\text{TEST}}$ is asserted, all output pins except OSCX2 and RTCX2 are TRI-STATE.

Table 1-9: Interrupt Control Pins

Symbol	Pins	Type	Function
NMI	55	I	**N**on-**M**askable **I**nterrupt. This active-high signal will generate a non-maskable interrupt to the CPU when it is active high. Normally this signal is used to indicate a serious system error.
$\overline{\text{INTA}}$	56	O	**INT**errupt **A**cknowledge. During each interrupt acknowledge cycle this signal will strobe low; it should be used by external cascaded interrupt controllers.
IRQ[7:0]	46, 47, 57, 58, 59, 60, 61, 62	I	**I**nterrupt **R**e**Q**uests. These inputs are either rising edge or low-level sensitive interrupt requests, depending on the configuration of the internal interrupt controllers. These interrupt requests may also be programmed to support externally cascaded interrupt controller(s). The IRQ pins are also used to select a particular test in test mode.

Table 1-10: Real Time Clock Pins

Symbol	Pins	Type	Function
RTCX1	38	I	**R**eal **T**ime **C**lock crystal oscillator input: 32kHz crystal.
RTCX2	40	O	**R**eal **T**ime **C**lock crystal oscillator output: 32kHz crystal.
Vbat	41	I	External + battery input for real time clock.

Table 1-11: Oscillator Pins

Symbol	Pins	Type	Function
SYSCLK	73	O	**SYS**tem **CL**oc**K**. This clock output pin will either be driven with a signal half the frequency of the OSCX1 input clock frequency or the CPU's clock frequency, which is determined in the Power Management Control Register 1. The source selection for this signal is determined by bit 1 of the Power Management Control Register 3.
OSCX1	22	I	**OSC**illator Crystal **1** input. This pin should either be driven by a TTL oscillator or be connected to an external crystal circuit. This signal is the fundamental clock source for all clocked elements in the **NS486SXL**, except the Real-Time Clock, which has its own crystal pins.
OSCX2	23	O	**OSC**illator Crystal **2** output. This is the output side of the **NS486SXL** on-chip circuitry provided to support an external crystal circuit. If a TTL oscillator drives OSCX1, this pin should be a no connect.

Table 1-12: 16550 UART Pins

Symbol	Pins	Type	Function
Tx	33	O	UART **T**ransmit data. In IrDA and HP-SIR mode this pin is the UART output encoded for the serial infrared link. Otherwise it is the transmit output of the 16550 UART.
Rx	34	I	UART **R**eceive data. In IrDA and HP-SIR mode this pin is routed through the serial infrared decoder. Otherwise, it is the receive input to the 16550.
UCLK	35	O	**U**art **CL**oc**K**. Output of programmable rate UART/MODEM clock. Typically used for the Infrared Modulator
$\overline{\text{RTS}}$	28	O	**R**equest **T**o **S**end. When low, this signal informs the MODEM or data set that the UART is ready to exchange data. The $\overline{\text{RTS}}$ output signal can be set to an active low by programming bit 2 (RTS) of the MODEM Control Register. A Master Reset operation sets this signal to its inactive (high) state. Loop mode operation holds this signal in its inactive state.
$\overline{\text{DSR}}$	29	I	**D**ata **S**et **R**eady. When low, it indicates that the MODEM or data set is ready to link with the UART. The $\overline{\text{DSR}}$ signal is a MODEM status input whose condition can be tested by reading bit 5 (DSR) of the MODEM Status Register. Bit 5 is the complement of the $\overline{\text{DSR}}$ signal. Bit 1 (DDSR) of the MODEM Status Register indicates whether the $\overline{\text{DSR}}$ input has changed state since the previous reading of the MODEM Status Register. **Note:** Whenever the DSR bit of the MODEM Status Register changes state, an interrupt is generated if the MODEM Status Interrupt is enabled.
$\overline{\text{DTR}}$	30	O	**D**ata **T**erminal **R**eady. When low, this signal informs the MODEM or data set that the UART is ready to establish a communications link. The $\overline{\text{DTR}}$ output signal can be set to an active low by programming bit 0 (DTR) of the MODEM Control Register to a high level. A Master Reset operation sets this signal to its inactive (high) state. Loop mode operation holds this signal in its inactive state.
$\overline{\text{DCD}}$	25	I	**D**ata **C**arrier **D**etect. When low, this input signal indicates that the data carrier has been detected by the MODEM or data set. The $\overline{\text{DCD}}$ signal is a MODEM status input whose condition can be tested by reading bit 7 (DCD) of the MODEM Status Register. Bit 7 is the complement of the $\overline{\text{DCD}}$ signal. Bit 3 (DDCD) of the MODEM Status Register indicates whether the $\overline{\text{DCD}}$ input has changed state since the previous reading of the MODEM Status Register. $\overline{\text{DCD}}$ has no effect on the receiver. **Note:** Whenever the DCD bit of the MODEM Status Register changes state, an interrupt is generated if the MODEM Status Interrupt is enabled.
$\overline{\text{CTS}}$	26	I	**C**lear **T**o **S**end. When low, this input signal indicates that the MODEM or data set is ready to exchange data. The $\overline{\text{CTS}}$ signal is a MODEM status input whose conditions can be tested by reading bit 4 (CTS) of the MODEM Status Register. Bit 4 is the complement of the $\overline{\text{CTS}}$ signal. Bit 0 (DCTS) of the MODEM Status Register indicates whether the $\overline{\text{CTS}}$ input has changed state since the previous reading of the MODEM Status Register. $\overline{\text{CTS}}$ has no effect on the Transmitter. **Note:** Whenever the CTS bit of the MODEM Status Register changes state, an interrupt is generated if the MODEM Status Interrupt is enabled.

$\overline{RI}$	27	I	**R**ing **I**ndicator. When low, this input signal indicates that a telephone ringing signal has been received by the MODEM or data set. The $\overline{RI}$ signal is a MODEM status input whose condition can be tested by reading bit 6 (RI) of the MODEM Status Register. Bit 6 is the complement of the $\overline{RI}$ signal. Bit 2 (TERI) of the MODEM Status Register indicates whether the $\overline{RI}$ input signal has changed from a low to high state since the previous reading of the MODEM Status Register. **Note:** Whenever the CTS bit of the MODEM Status Register changes state, an interrupt is generated if the MODEM Status Interrupt is enabled. **Note:** Whenever the RI bit of the MODEM Status Register changes from a high to a low state, an interrupt is generated if the MODEM Status Interrupt is enabled.

Table 1-13: Timer Pins

Symbol	Pins	Type	Function
T0	31	I/O	Programmable **T**imer pin **0**. This Bidirectional pin may be selected to control one of the following four functions via bits 1-0 of the Timer I/O Control Register: 1) The GATE input into Timer 0. 2) The GATE input into Timer 1. 3) The OUT output from Timer 0. 4) The CLK input into Timer 1.
T1	32	I/O	Programmable **T**imer pin **1**. This Bidirectional pin may be selected to control one of the following four functions via bits 3-2 of the Timer I/O Control Register: 1) The GATE input into Timer 0. 2) The GATE input into Timer 1. 3) The OUT output from Timer 1. 4) The CLK input into Timer 0.

Table 1-14: 3-Wire Serial I/O Pins

Symbol	Pins	Type	Function
SO	18	O	The **S**erial data **O**utput signal for MICROWIRE.
SI	19	I	The **S**erial data **I**nput signal for MICROWIRE or the serial data I/O for Access.bus.
SCLK	20	O	The **S**erial **CL**oc**K** signal for MICROWIRE and Access.bus.

Note: For MICROWIRE Slave Mode, a pin must be selected to be the Chip Select Input.

Table 1-15: Summary of Reconfigurable I/O Pins

Symbol	Pins	Type	Original Function
$\overline{\text{DRV}}$	54	I/O	Auxiliary Processor
$\overline{\text{EACK}}$	53	I/O	Auxiliary Processor
$\overline{\text{EREQ}}$	52	I/O	Auxiliary Processor
SCLK	20	I/O	Microwire/Access.bus
SI	19	I/O	Microwire/Access.bus
SO	18	I/O	Microwire/Access.bus
T1	32	I/O	Timer
T0	31	I/O	Timer
1RQ7	46	I/O	Interrupt Controller
1RQ6	47	I/O	Interrupt Controller
1RQ5	57	I/O	Interrupt Controller
1RQ4	58	I/O	Interrupt Controller
1RQ3	59	I/O	Interrupt Controller
1RQ2	60	I/O	Interrupt Controller
1RQ1	61	I/O	Interrupt Controller
1RQ0	62	I/O	Interrupt Controller
$\overline{\text{DTR}}$	30	I/O	UART
$\overline{\text{DSR}}$	29	I/O	UART
$\overline{\text{RTS}}$	28	I/O	UART
$\overline{\text{RI}}$	27	I/O	UART
$\overline{\text{CTS}}$	26	I/O	UART
$\overline{\text{DCD}}$	25	I/O	UART
RX	34	I/O	UART
UCLK	35	I/O	UART
$\overline{\text{CS[4]}}$	43	I/O	Chip Select 4
$\overline{\text{CS[3]}}$	44	I/O	Chip Select 3
$\overline{\text{CS[2]}}$	45	I/O	Chip Select 2
$\overline{\text{CS[1]}}$	63	I/O	Chip Select 1

These 28 pins, typically used for various I/O peripheral purposes, as defined in the above tables, can be reconfigured for use as general purpose I/O pins.

This Page Intentionally Left Blank.

2.0 System Overview

2.1 NS486SXL System Overview

The **NS486SXL** is a highly integrated embedded system controller. It includes an Intel486-class 32-bit processor, all resources required for the System Service Elements of a Real-Time Executive, and a selection of key I/O peripherals. This “system-on-a-chip” is ideal for implementing a wide variety of embedded applications.

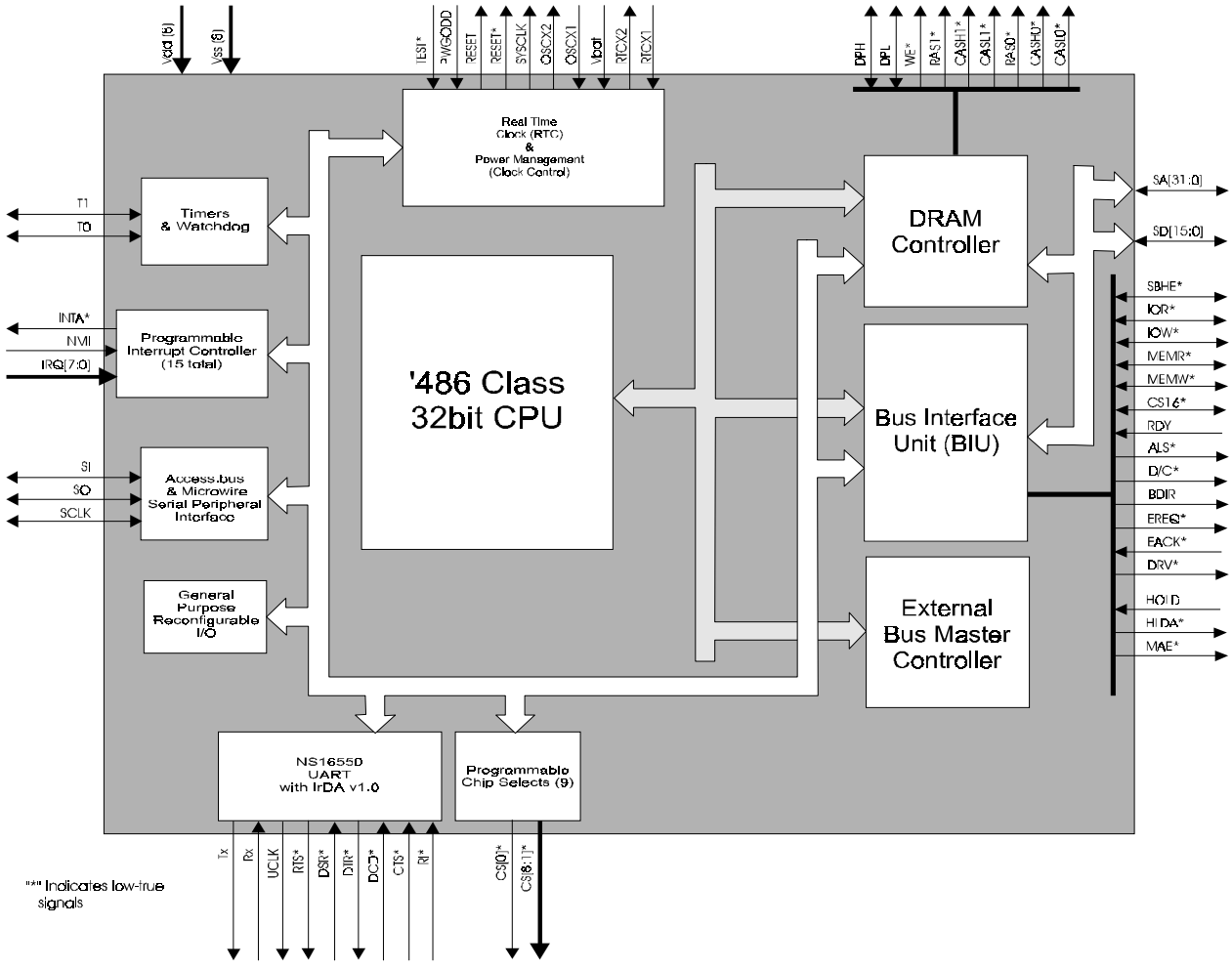


Figure 2-1 NS486SXL Internal Resource to Pins Map

The 32-bit processor core executes all of the Intel486 instructions with a similar number of clocks per instruction. An on-board 1Kbyte instruction cache provides for efficient execution from ROM. Intel486 debug features are supported. The processor has been optimized for operating system kernels such as VRTX, VxWorks, pSOS+ and QNX. These environments only need the '486 protected mode operation (no real mode or virtual 8086 support), flat or linear memory addressing (no virtual memory paging), and floating point execution in software only (no co-processor interface).

In fact, the **NS486SXL** includes all of the System Service Elements required by a typical kernel, including an efficient DRAM controller that supports page-mode DRAMs for data cache-like performance; three timer channels (including one configured as a protected WATCHDOG Timer); two programmable 8259 interrupt controllers provide 15 on-chip interrupt sources; an industry standard real time clock and calendar (RTC) with battery backup; and support for comprehensive power management schemes.

In addition, the **NS486SXL** also incorporates the key I/O peripherals required for implementing a wide variety of embedded applications: an industry standard high-performance NS16550-compatible UART with HP-SIR and IrDA v1.0 infrared option, an 8254 timer, and a general purpose 2- or 3-wire synchronous serial interface for easy interface to low-cost EEPROMs and other serial peripherals. System expansion is supported with nine programmable Chip Select (CS) signals and a generic ISA-type bus interface for external devices and memory.

Certain I/O lines not being used by disabled peripherals can be reconfigured for use as general purpose bi-directional I/O lines (up to 28 pins)**See "Reconfigurable I/O (RIO)" on page 107.** This gives the designer maximum flexibility in designing various systems using the **NS486SXL** device. It is expected that an **NS486SXL** system will minimally include the **NS486SXL** system controller with on-board processor and I/O devices, boot ROM, and working RAM memory. Many applications will not require any additional I/O support.

Finally, the **NS486SXL** implements a very flexible power management scheme that permits selective control of individual I/O subsystems, with varying levels of power consumption.

NS486SXL provides a cost-effective hardware platform for the design and implementation of a wide range of office automation and communication systems. With its powerful embedded '486-class processor, comprehensive set of on-chip peripheral controllers, flexible power management structure and reconfigurable I/O lines, **NS486SXL** makes possible a variety of end-user systems based on the same hardware. Because of its optimized design and on-board resources, a very cost effective system can be achieved.

2.2 32-bit Processor Core

The NS486SXL processor core is an implementation of the protected mode '486 instruction set architecture, optimized using a RISC-like design philosophy for embedded applications. Using this approach, the most frequently used instructions are optimized, and on an average execute in a lower number of clock cycles than a '486.

The NS486SXL features a three stage pipeline, efficient instruction prefetching mechanism, and single cycle instruction decoding for most instructions. Additionally, a 1K byte instruction cache and single cycle DRAM access provide higher memory performance than a larger unified cache implementation.

The NS486SXL processor provides the same programming model and register set as the standard '486 except that real mode, virtual memory, and floating point support have been eliminated. These features have little or no impact in embedded applications and save significant silicon real estate. At reset, unlike the standard '486, the NS486SXL starts up in protected mode instead of real mode. All '486 instructions appropriate to protected mode and our hardware configuration are supported, including debug instructions.

The NS486SXL is initially available to run 25 MHz at 5 Volts. The processor clock is obtained by dividing the crystal frequency by two. For example, a 25 MHz NS486SXL runs with a 50 MHz crystal oscillator as the master clock.

As a result of our innovative design, the NS486SXL achieves performance equivalent to a standard '486 with less circuitry. This translates into reduced power consumption and a lower overall system cost. It also makes the NS486SXL ideal for "green" systems and battery operated systems.

2.3 System Service Elements

The **NS486SXL** controller provides the basic hardware resources required for the O/S-defined System Service Elements. These include a DRAM controller,

programmable interval timer, a protected WATCHDOG timer, a programmable interrupt controller, a real-time clock and calendar, and comprehensive power management features.

2.3.1 DRAM Controller

The **NS486SXL** DRAM controller supports one or two adjustable-sized banks of dynamic RAM using a 16-bit data path. Support is provided for byte parity (if desired), requiring the DRAM banks to be 18-bits wide when parity is enabled. Banks can be up to 8 Mbytes in size. The DRAM controller supports page mode read and write operations and can also support both byte and word accesses. All access control signals for read, write and parity checking are generated as well as an automatic and programmable CAS-before-RAS refresh. If self-refresh DRAMs are used, refresh can be disabled, saving power.

NS486SXL provides flexible support for use of a number of different DRAM configurations, using popular DRAM devices. Access is optimized for fast page mode DRAMs, and they will provide the highest performance with contiguous data. When accessing data bytes or words in the same DRAM page, the data access is in one cycle. This performance provides fast data access times without the overhead of a separate data cache. Page sizes can be 512, 1024, 2048 or 4096 bytes. Flexibility for DRAM timing is provided through programming of the DRAM controller registers: 3 or 4 cycle page miss accesses and extended CAS cycles can be selected.

Memory bank 0 starts at address 0h; memory bank 1 can start at any address in the 128 Mbyte address map that is a multiple of its size.

2.3.2 Programmable Interval Timer

The **NS486SXL** programmable interval timer is compatible with the Intel 8254 programmable interval timer and contains three identical timers (CH0-CH2). CH0 and CH1 can be used to generate accurate timing delays under software control. CH2 may be configured to provide a WATCHDOG timer function.

2.3.3 WATCHDOG Timer

The **NS486SXL** WATCHDOG timer, CH2, is a protected 16-bit timer that can be used to prevent system "lockups or hangups." It uses a 1 KHz clock generated by the on-chip real-time clock circuit. If the WATCHDOG timer is enabled and times out, a reset or interrupt will be generated allowing graceful recovery from an unexpected system lockup.

2.3.4 Interrupt Controller

The **NS486SXL** interrupt controller consists of two cascaded programmable interrupt controllers that are compatible with the Intel 8259A Programmable Interrupt Controller. They provide a total of 15 (out of 16) programmable interrupts. Three interrupts are reserved for a real time clock-tick interrupt, a real time clock interrupt request, and a cascade interrupt channel. The remaining 13 interrupts can be used by internal or external sources. Additional external interrupt controllers can be cascaded as well.

2.3.5 Real Time Clock/Calendar

The **NS486SXL** Real Time Clock/Calendar is a low power clock that provides a time-of-day clock and 100-year calendar with alarm features and battery operation. Time is kept in BCD or binary format. It includes 50 bytes of general purpose CMOS RAM and 3 maskable interrupt sources. It is compatible with the DS1287 and MC146818 RTC/Calendar devices, except for the general purpose memory size.

2.3.6 Power Management Features

The **NS486SXL** power management structure includes a number of power saving mechanisms that can be combined to achieve comprehensive power savings under a variety of system conditions. First of all, the core processor power consumption can be controlled by varying the processor/system clock frequency. The internal CPU clock can be divided by 4, 8, 16, 32 or 64. In addition, in idle mode, the internal processor clock will be disabled. Finally, if an external crystal oscillator circuit is being used, it can be disabled. For maximum power savings, all internal clocks can be disabled (except for the real-time clock oscillator).

The clocks of the on-board peripherals can be individually or globally controlled. By setting bits in the power management control registers, the internal clocks to the three-wire interface, the timer, the DRAM controller, and the UART can be disabled.

In addition to these internal clocks, the external SYSCLK can be disabled via a bit in the power management control registers.

Using various combinations of these power saving controls with the **NS486SXL** controller will result in excellent programmable power management for any application.

2.4 NS486SXL System Bus

The **NS486SXL** system bus provides the interface to off-chip peripherals and memory. It offers an ISA-compatible interface and is therefore capable of directly interfacing to many ISA peripheral control devices. The interface is accomplished through the Bus Interface Unit (BIU). The BIU generates all of the access signals for both internal and external peripherals and memory. Depending upon whether the access is to internal peripherals, external peripherals or external memory, the BIU generates the timing and control signals to access those resources. The BIU is designed to support a glueless interface to many ISA-type peripherals.

For debug purposes, the **NS486SXL** can be set to generate external bus cycles at the same time as an internal peripheral access takes place. This gives logic analyzers or other debug tools the ability to track and capture internal peripheral accesses.

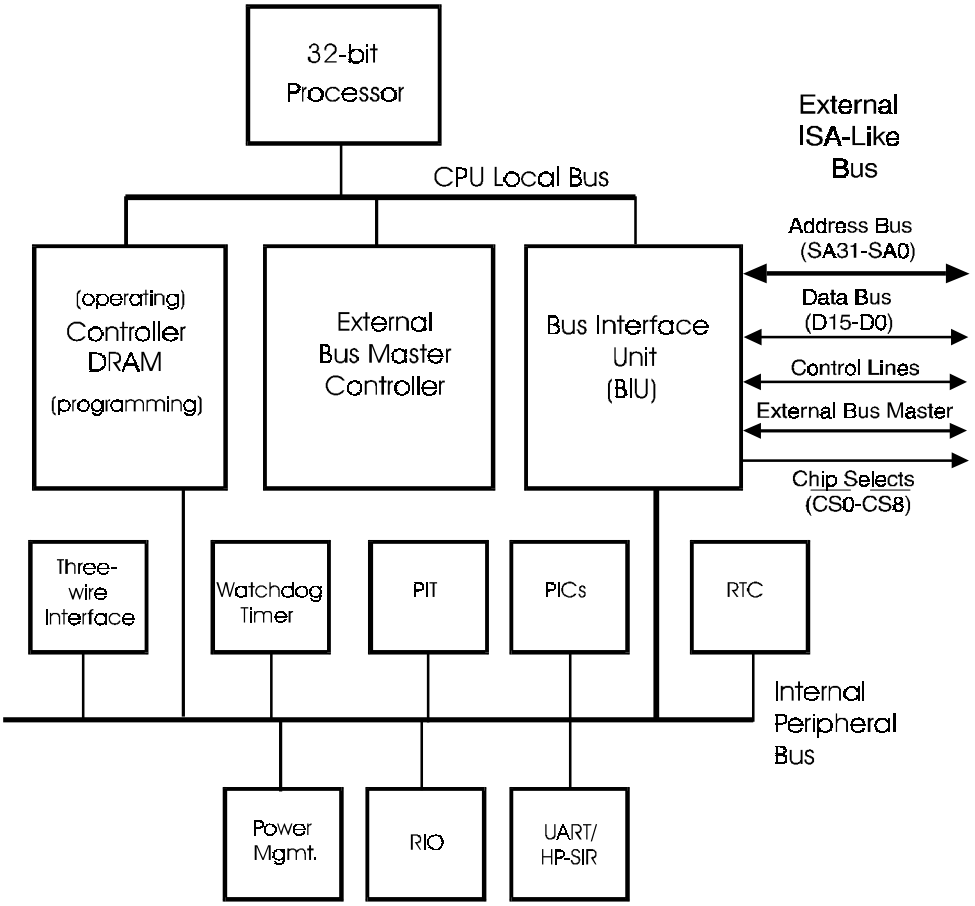


Figure 2-1 NS486SXL Internal Busses

Access to internal peripherals is accomplished in three CPU T-states (clock cycles). The fastest access to off-chip I/O is also three T-states. When accessing off-chip memory and I/O, wait state generation is accomplished through a combination of **NS486SXL** chip select logic and off-chip peripheral feedback signals.

The ISA-like bus on the **NS486SXL** also supports External Bus Masters. This feature allows external processors or I/O Peripherals (and customer proprietary ASICs) with built-in DMA controllers to read and write System DRAM supported by the 'SXL DRAM Controller. External Masters can also access any internal or external peripherals or memory as well. The external master address must be tri-stateable (through external address transceivers if necessary) in order to support external master access to the DRAM.

Finally, the Bus Interface Unit also provides signals to indicate bus activity and control (optional) data bus transceiver direction. The bus direction signal, and chip selects continue to operate correctly during external master accesses.

2.5 Other On-board Peripherals

In addition to those peripherals and system control elements needed for System Service Elements, the **NS486SXL** also includes a number of I/O controllers and resources that make implementing a complete embedded system possible with just a single-chip **NS486SXL** controller. These include a serial UART port, and a MICROWIRE or Access.bus synchronous serial bus interface.

2.5.1 Reconfigurable I/O Lines

The **NS486SXL** supports reconfigurable I/O. For example, if the UART, interrupts, or other functions are not being used, the I/O pins associated with them can be reconfigured as general purpose bidirectional I/O pins. Up to 28 pins can be reconfigured for this purpose. This capability makes the **NS486SXL** extremely versatile and ideal for supporting different end product configurations with a single **NS486SXL** device.

2.5.2 MICROWIRE/Access.bus Interface

The **NS486SXL** MICROWIRE/Access.bus interface provides for full support of either the three-wire MICROWIRE or the two-wire Access.bus serial interfaces. MICROWIRE has an alternate clock phasing option that supports the SPI bus protocol as well. These industry standard interfaces permit easy interfacing to a wide range of low-cost specialty memories and I/O devices. These include EEPROMs, SRAMs, timers, clock chips, A/D converters, D/A converters, and peripheral device drivers.

2.5.3 UART Serial Port

The **NS486SXL** UART provides complete NS16550 (PC standard) serial communications port compatibility including the performance enhancing 16-byte deep FIFO. It performs serial-to-parallel conversion from external devices to the **NS486SXL** and parallel-to-serial conversion from the **NS486SXL** to external peripherals. Full modem control can be supported.

A serial IrDA v1.0 and HP-SIR (infrared) mode is also supported, making possible low-cost wireless communications between an **NS486SXL**-based system and other wireless infrared systems.

2.6 ICE Support

National Semiconductor has worked closely with Microtek International to provide hardware in-circuit emulator support for the **NS486SXL**. The Microtek product (PowerPack® EA-NS486) uses a special bondout version of the **NS486SXL** to deliver a full-featured hardware emulator that is capable of tracing on chip activity, including peripheral interrupt and I/O activity. The emulator runs at full speed, and supports overlay memory and multiple triggers.

PowerPack is a registered trademark of Microtek International.

2.7 Other Issues

NS486SXL provides a comprehensive set of on-board peripherals. Also, it is designed to easily interface to external peripherals. In addition to this ISA-like bus which supports ISA-compatible peripherals, the **NS486SXL** provides an interface to an external master with a shared memory space. The external master or auxiliary processor interface allows low cost interfacing to shared external memory belonging to other external masters (including another **NS486SXL** controller).

To program the resources of the **NS486SXL**, a set of internal control registers exists. These registers provide precise control over all internal resources and the setup of external **NS486SXL** control signals. It is the designer's responsibility to ensure the proper initialization of the registers in this I/O map.

In addition, the **NS486SXL** core processor itself requires several descriptor tables and initialization parameters that must be set by user-written start-up software.

The **NS486SXL** is designed from the ground up for optimum price/performance in embedded systems. This makes the **NS486SXL** the logical choice as the base hardware platform for executing an embedded operating system kernel such as those available from Microtec International, Wind River, ISI, QNX, and many others. Any Operating System or Real-Time Executive that will operate in a segmented or flat memory model protect mode environment is a suitable complement to the **NS486SXL**.

Also, there are many third party tool sets that will allow an executable application to be built to run directly on the target hardware without an O/S environment.

3.0 Programming the NS486SXL

3.1 Overview

NS486SXL resources are setup and controlled via a set of control registers located in the I/O map. It is the system designer's responsibility to configure the various I/O devices and other elements of the NS486SXL hardware through startup software routines.

Like other '486-type processors, the NS486SXL has a 64Kbyte I/O address space map. In the NS486SXL, the I/O address space addresses physical memory and I/O ports are accessed using the IN and OUT I/O instructions.

For more information on the NS486SXL core processor and its addressing modes see Section 5, CPU Overview.

As shown in the figure, the NS486SXL I/O map is logically divided into several segments, each containing various control registers for the hardware elements of the NS486SXL single-chip system. To provide some continuity for the programmer familiar with the standard personal computer DOS/MS-DOS/ Windows environment, key address locations for the UART, interrupts, real time clock, etc., have been retained from that I/O model wherever possible. Where continuity is not possible, or new resources are available, new locations are used.

The complete NS486SXL address map of the control registers is shown in the tables on the following pages.

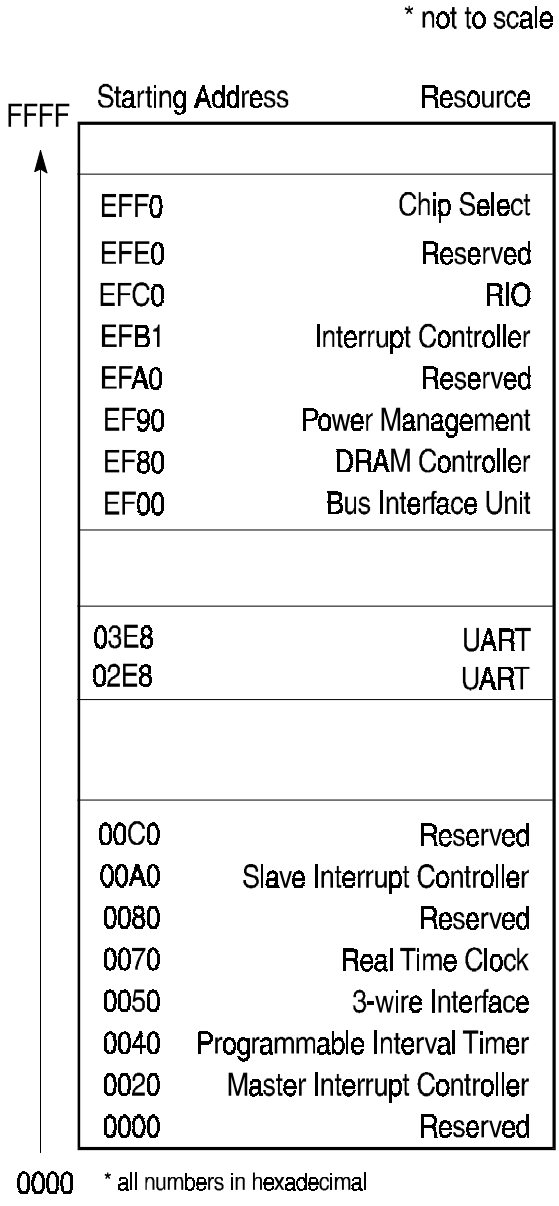


Figure 3-1 NS486SXL I/O Utilization

3.2 I/O Address Map

The following tables list all of the registers within the NS486SXL, their respective I/O address locations and where more information about them can be found.

Table 3-1: I/O Address Map

I/O Address	Register Name	Reference Section
0020h	Interrupt Controller (I/O Port 0)	Master Interrupt Controller (7-0)
0021h	Interrupt Controller (I/O Port 1)	Master Interrupt Controller (7-0)
0040h	Counter 0 Register	Prog. Interval Timer
0041h	Counter 1 Register	Prog. Interval Timer
0042h	Counter 2 Register	Prog. Interval Timer
0043h	Control Word Register	Prog. Interval Timer
0044h	Timer I/O Control Register	Prog. Interval Timer
0045h	Timer Clock Register	Prog. Interval Timer
0046h	WATCHDOG Retrigger	WATCHDOG Timer
0047h	WATCHDOG Timer Control Register	WATCHDOG Timer
0050h	System Control Register	3-Wire Interface
0051h	Serial Input/Output Register	3-Wire Interface
0052h	MICROWIRE Control Register	3-Wire Interface
0053h	Serial Input/Output DATA Register	3-Wire Interface
0054h	Access.bus Status Register	3-Wire Interface
0055h	Access.bus Control Register	3-Wire Interface
0056h	Own Address Register	3-Wire Interface
0057h	Access.Bus Control Register 2	3-Wire Interface
0070h	Real Time Clock Index Register	Real Time Clock
0071h	Real Time Clock Data Port	Real Time Clock
00A0h	Interrupt Controller (I/O Port 0)	Slave Interrupt Controller (15-8)
00A1h	Interrupt Controller (I/O Port 1)	Slave Interrupt Controller (15-8)
02E8h - 02EFh	COM4 *(option)*	UART
02F8h - 02FFh	COM2 *(option)*	UART
03E8h - 03EFh	COM3 *(option)*	UART
03F8h - 03FFh	COM1 *(default UART location)*	UART
EF00h	Bus Interface Control Register 1	Bus Interface Unit
EF01h	Bus Interface Control Register 2	Bus Interface Unit
EF02h	Chip Select Enable Register	Bus Interface Unit

Table 3-1: I/O Address Map

I/O Address	Register Name	Reference Section
EF03h	Chip Select Type Register	Bus Interface Unit
EF04h-EF07h	Chip Select Base Address Register 1	Bus Interface Unit
EF08h-EF0Bh	Chip Select Base Address Register 2	Bus Interface Unit
EF0Ch-EF0Fh	Chip Select Base Address Register 3	Bus Interface Unit
EF10h-EF13h	Chip Select Base Address Register 4	Bus Interface Unit
EF14h-EF17h	Chip Select Base Address Register 5	Bus Interface Unit
EF18h-EF1Bh	Chip Select Base Address Register 6	Bus Interface Unit
EF1Ch-EF1Fh	Chip Select Base Address Register 7	Bus Interface Unit
EF20h-EF23h	Chip Select Base Address Register 8	Bus Interface Unit
EF24h-EF27h	Chip Select Address Mask Register 1	Bus Interface Unit
EF28h-EF2Bh	Chip Select Address Mask Register 2	Bus Interface Unit
EF2Ch-EF2Fh	Chip Select Address Mask Register 3	Bus Interface Unit
EF30h-EF33h	Chip Select Address Mask Register 4	Bus Interface Unit
EF34h-EF37h	Chip Select Address Mask Register 5	Bus Interface Unit
EF38h-EF3Bh	Chip Select Address Mask Register 6	Bus Interface Unit
EF3Ch-EF3Fh	Chip Select Address Mask Register 7	Bus Interface Unit
EF40h-EF43h	Chip Select Address Mask Register 8	Bus Interface Unit
EF44h	ROM BIOS Selection Register	Bus Interface Unit
EF45h	External Chip Select Selection Register 1	Bus Interface Unit
EF46h	External Chip Select Selection Register 2	Bus Interface Unit
EF47h	External Chip Select Selection Register 3	Bus Interface Unit
EF48h	External Chip Select Selection Register 4	Bus Interface Unit
EF49h	External Chip Select Selection Register 5	Bus Interface Unit
EF4Ah	External Chip Select Selection Register 6	Bus Interface Unit
EF4Bh	External Chip Select Selection Register 7	Bus Interface Unit
EF4Ch	External Chip Select Selection Register 8	Bus Interface Unit
EF50h	Chip Select Access Time Register 1	Bus Interface Unit
EF51h	Chip Select Access Time Register 2	Bus Interface Unit
EF52h	Chip Select Access Time Register 3	Bus Interface Unit
EF53h	Chip Select Access Time Register 4	Bus Interface Unit
EF54h	ROM BIOS Access Time Register	Bus Interface Unit
EF55h	Catchable Chip Selects Register	Bus Interface Unit
EF56h	Auxiliary Processor Chip Select Selection Register	Bus Interface Unit
EF57h	Programmed 16-bit Logical Chip Select Register	Bus Interface Unit
EF70h	UART Clock Divisor Register	UART

Table 3-1: I/O Address Map

I/O Address	Register Name	Reference Section
EF71h	Modem Signal Control Register	UART
EF80h - EF81h	DRAM Control Register	DRAM Controller
EF82h - EF83h	Refresh Rate Register	DRAM Controller
EF84h - EF85h	$\overline{RAS}$ Timeout Register	DRAM Controller
EF86h - EF87h	DRAM Status Register	DRAM Controller
EF88h	DRAM Bank Size Register	DRAM Controller
EF89h	Bank 1 Address Register	DRAM Controller
EF8Ah	Bank 0 Mask Register	DRAM Controller
EF8Bh	Bank 1 Mask Register	DRAM Controller
EF90h	Power Management Register 1	Power Mgmt.
EF91h	Power Management Register 2	Power Mgmt.
EF92h	Power Management Register 3	Power Mgmt.
EF93h	Power Management Register 4	Power Mgmt.
EFB0h	Misc. Interrupt Selection Register 1	Interrupt Controller
EFB1h	Internal IRQ1 Source Selection Register	Interrupt Controller
EFB2h	Misc. Interrupt Selection Register 2	Interrupt Controller
EFB3h	Internal IRQ3 Source Selection Register	Interrupt Controller
EFB4h	Internal IRQ4 Source Selection Register	Interrupt Controller
EFB5h	Internal IRQ5 Source Selection Register	Interrupt Controller
EFB6h	Internal IRQ6 Source Selection Register	Interrupt Controller
EFB7h	Internal IRQ7 Source Selection Register	Interrupt Controller
EFB8h	Misc. Interrupt Selection Register 3	Interrupt Controller
EFB9h	Internal IRQ9 Source Selection Register	Interrupt Controller
EFBAh	Internal IRQ10 Source Selection Register	Interrupt Controller
EFBBh	Internal IRQ11 Source Selection Register	Interrupt Controller
EFBCh	Internal IRQ12 Source Selection Register	Interrupt Controller
EFBDh	Internal IRQ13 Source Selection Register	Interrupt Controller
EFBEh	Internal IRQ14 Source Selection Register	Interrupt Controller
EFBFh	Internal IRQ15 Source Selection Register	Interrupt Controller
EFC0h	Reconfigurable I/O Control Register	RIO
EFC1h	MICROWIRE Slave Mode Chip Select Register	RIO
EFC2h	Device ID Register	Bus Interface Unit
EFC3h	Device Revision Register	Bus Interface Unit
EFC4h - EFC7h	Data Direction Register	RIO
EFC8h - EFCBh	Data Port In Register	RIO

Table 3-1: I/O Address Map

I/O Address	Register Name	Reference Section
EFCCh - EFCFh	Data Port Out Register	RIO
EFF0h - EFF0h	Chip Select Local Device Register	Bus Interface Unit

3.3 System Service Elements

NS486SXL on-chip System Service Elements include the DRAM controller, programmable interval timers, a protected watchdog timer, programmable interrupt controllers, a real-time clock and calendar, and power management logic.

3.3.1 The DRAM Controller

NS486SXL's DRAM controller is designed specifically for fast-page mode DRAMs with CAS-before-RAS refresh capability. Configuration and operation is controlled by the settings in eight DRAM Controller registers in the NS486SXL I/O map. (See Table 3-1 on page 24). The user must set the page size of the DRAMs, the size of the memory banks and where bank 1 will be located, whether parity will be used, and if so, whether a parity error will generate an NMI interrupt, and finally, whether the DRAMs use 3 or 4 clock cycles on a page miss access. (A typical configuration with a CPU clock of 25 MHz and a 3 cycle page miss would require 60 ns DRAMs.)

One or two banks of up to 8 Mbytes each of Dynamic RAM are supported. Bank 0 must be placed at the beginning of the flat memory address space of the NS486SXL processor. The banks can have different page sizes and memory sizes. The NS486SXL DRAM controller supports page mode read and write operations and both byte and word accesses.

All access control signals for read, write and parity checking are generated, as well as an automatic and programmable CAS-before-RAS refresh. If self-refresh DRAMs are used, refresh is not generated, saving power.

A number of different DRAM configurations using popular DRAM devices are supported. Page sizes can be 512, 1024, 2048, or 4096 bytes.

Memory bank 0 must start at address 0h; memory bank 1 must start at an address that is a multiple of that bank's size. For example, a 4 Mbyte bank must start at 400000h, 800000h, C00000h, etc. Memory bank address ranges cannot overlap.

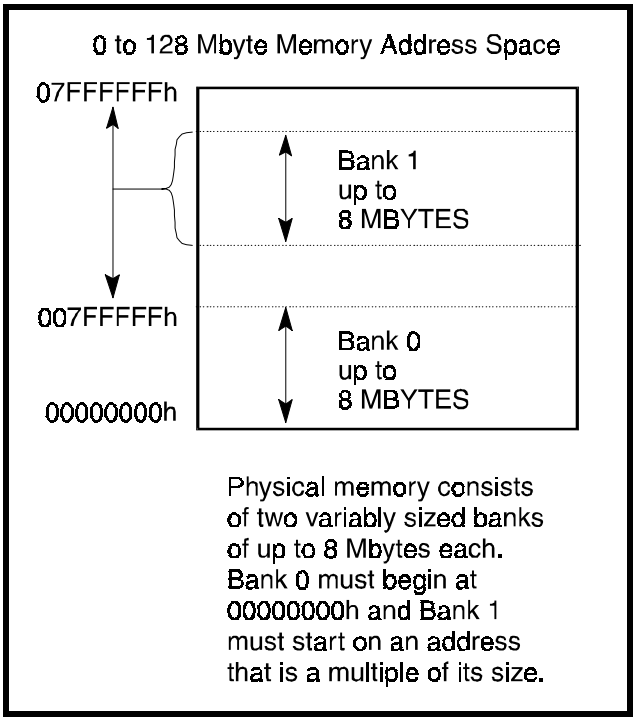


Figure 3-2 Memory Addressing

The NS486SXL DRAM controller supports five different DRAM Bank Sizes (0.5, 1, 2, 4, and 8 MByte) and five different DRAM Device Page Sizes (8, 9, 10, 11, and 12 Address Bits) independant of each other to provide the maximum flexibility for a wide range of new and old DRAM devices and memory modules.

Self-refresh mode for low-power operation can be used if the specific DRAM type supports it. Exiting the self-refresh mode is accomplished automatically when a hardware interrupt is generated, or by writing the appropriate control register bit (DRAM Control Register, bit 3=0).

Hardware Interconnection

NS486SXL address signals SA1... SA11 connect to Logical Address signals A0... A10 on the DRAM devices. NS486SXL control signals CASH0/CASL0 and CASH1/CASL1 are used to select upper and lower byte access. The NS486SXL generates separate CAS signals for each byte. If 16-bit wide DRAM chips are used, they should have two CAS signals.

The DRAM Controller can drive up to a maximum load of 100 pF on the SA12-1 signal pins. (**Note:** A system that uses the external bus for more than just DRAM will require signal buffering.) The DRAM Controller can drive a maximum of 30 pF on the $\overline{CAS}$ signals, 60 pF on the $\overline{RAS}$ signals and 120 pF on the $\overline{WE}$ signal. Typically, a system can support a maximum of 12 DRAM devices at a time, 6 per bank. In this case, the DRAMs must have a 4-bit or wider data bus (except for the parity bit if implemented separately).

When running the CPU at 25 MHz, the DRAM Controller will operate with most 60ns DRAMs using 3 CPU cycles for a page miss and one cycle for a page hit (**Note**: Some 60ns DRAMs have a slow CAS access time). Slower DRAMs may be used by using a slower clock or by using Four Cycle Miss Mode (DRAM Control Register, bit 7 =1).

3.3.1.1 Reset Condition

A system reset will place the DRAM Controller into an idle state. Following reset, the DRAM controller will wait for at least 100 µsec or until the DRAM Enable bit (DRAM Control Register, I/O address EF80-81h, bit 0) is programmed to a 1, whichever occurs later. After the 100 µsec delay, and with the DRAM Enable bit set, the DRAM Controller will generate eight CAS-before-RAS refresh cycles. At the end of the eighth refresh cycle, the DRAM Initialization Complete bit (DRAM Status Register, address EF86-87, bit 0) is set to a 1, indicating that the DRAM may now be read and written. The user program can check this bit to see when the DRAM is ready. Note that if during operation, the DRAM Enable bit (DRAM Control Register, I/O address EF80-81h, bit 0) is set to 0, the DRAM controller will not stop immediately. It will wait until the end of the next refresh cycle. However, subsequent re-enabling of the DRAM controller will occur immediately when a 1 is written to the DRAM Enable bit.

3.3.1.2 DRAM Parity Checking

The Odd-Parity generation and checking of the DRAM memory may be enabled or disabled by the Parity Enable bit. Internally, the Parity Error interrupt is connected to the CPU NMI logic for system intervention and error handling. When the Parity Error NMI Enable bit (DRAM Control Register, bit 14) is set to one, a parity error will generate an NMI to the CPU.

When the CPU gets an NMI, it can check to see if it came from the DRAM by checking the bits in the DRAM Status Register at location EF86-87h. Note that even if NMI is not enabled, the CPU can poll the Status Register to detect a parity error.

3.3.1.3 Refresh

Standard CAS-before-RAS refresh mode is supported. The refresh rate is programmable via the Refresh Rate Register (I/O address EF82h).

By programming the Self-Refresh Mode bit, (DRAM Control Register, I/O address EF80h, bit 3), to a 1, the controller will put the DRAMs into Self-Refresh mode by initiating a $\overline{CAS}$-before-$\overline{RAS}$ refresh cycle and holding both $\overline{CAS}$ and $\overline{RAS}$ low for the duration of the power-down mode. To exit Self-Refresh mode, either a 0 can be written to the Self-Refresh Mode bit, or a system interrupt can force the DRAM Controller out of Self-Refresh Mode. Exiting Self-Refresh mode will only provide one $\overline{CAS}$-before-$\overline{RAS}$ cycle. The DRAMs must be $\overline{CAS}$-before-$\overline{RAS}$ type, they must not require burst-refresh. Self-refresh DRAMs must be able to come out of refresh with a single $\overline{CAS}$ before $\overline{RAS}$ only. (**CAUTION:** Any DRAM access attempt while self-refresh is enabled will not complete and as a result will lock up the NS486SXL CPU. In this event, the WATCHDOG timeout is the only means of recovering.)

3.3.1.4 RAS Recovery (pre-charge) Rate

The DRAM Controller will force $\overline{RAS}$ high if it has remained low more than a certain period of time. The amount of time is programmed via the RAS Timeout Register. If the value written to this register is greater than the value written to the Refresh Rate Register,

then it is assumed that the programmed refresh period is less than or equal to the maximum RAS low period; therefore, the RAS Timeout Register will have no effect on the operation of the DRAM Controller.

3.3.1.5 The DRAM Registers

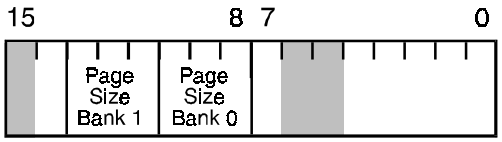
The DRAM controller is setup and its operation controlled by eight DRAM Controller registers in the NS486SXL I/O map. Four registers are 16-bit and four are 8-bit. They are:

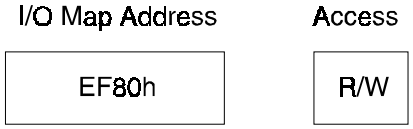
Name	I/O Address *	Size
DRAM Control Register	EF80-81	16 bits
Refresh Rate Register	EF82-83	16 bits
RAS Timeout Register	EF84-85	16 bits
DRAM Status Register	EF86-87	16 bits
DRAM Bank Size Register	EF88	8 bits
Bank 0 Mask Register	EF8A	8 bits
Bank 1 Mask Register	EF8B	8 bits
Bank 1 Address Register	EF89	8 bits
* addresses in hex		

It is important that the DRAM Controller be disabled (Bit 0 set to 0 in the DRAM Control Register, I/O address EF80h) before programming in order to avoid unpredictable or race conditions.

3.3.1.5.1 DRAM Control Register

This 16-bit register is used to control the operation of the DRAM Controller. The DRAM Control Register is written through I/O address EF80h.





Bit 15: Reserved

Bit 14: Parity Error NMI Enable
0 = Parity Error NMI Disabled (default)
1 = Parity Error NMI Enabled
Note: Bit 0 of the Miscellaneous Interrupt Selection Register 1 must be set in order for the NMI to reach the CPU. See Section 3.3.5.4.1 on page 78

Bits 13-11: DRAM Page Size, Bank 1.

Bit 13	Bit 12	Bit 11	DRAM Page Size (Address Bits)
0	0	0	8
0	0	1	9
0	1	0	10
0	1	1	11 (default)
1	0	0	12
1	0	1	Reserved
1	1	0	Reserved
1	1	1	Reserved

Bits 10-8: DRAM Page Size, Bank 0.

Bit 10	Bit 9	Bit 8	DRAM Page Size (Address Bits)
0	0	0	8
0	0	1	9
0	1	0	10
0	1	1	11 (default)

Bit 10	Bit 9	Bit 8	DRAM Page Size (Address Bits)
1	0	0	12
1	0	1	Reserved
1	1	0	Reserved
1	1	1	Reserved

Bit 7: Four Cycle Page Miss Mode
0 = Three Cycle Page Miss (default)
1 = Four Cycle Page Miss

Bits 6-5: Reserved

Bit 4: Extended CAS
0 = Normal CAS with 0.5 CPU clock low and high time (default)
1 = Extended CAS Low with 1.0 CPU clocks low and high time

Bit 3: Self-Refresh Mode
0 = Normal Refresh Mode (default). A system interrupt will also exit Self-Refresh Mode.
1 = Enter Self-Refresh Mode

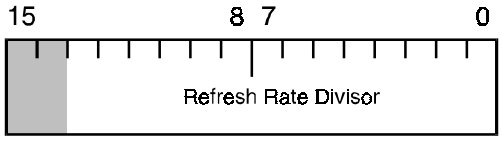
Bit 2: Parity Enable
0 = Parity Disabled (default)
1 = Parity Enabled

Bit 1: Bank 1 Present
0 = Bank 1 Empty (default)
1 = Bank 1 Present

Bit 0: DRAM Enable
0 = DRAM Disabled (default)
1 = DRAM Enabled

3.3.1.5.2 Refresh Rate Register

This 16-bit register is used to establish the frequency of refresh cycles. At reset, it is initialized to: 02EEh. The Refresh Rate Register is written through I/O address EF82h.



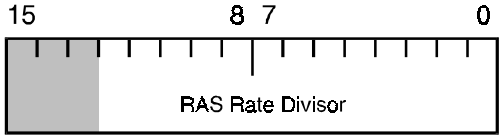
I/O Map Address	Access
EF82h	R/W

Bits 15-14: Reserved

Bits 13-0: Refresh Rate Divisor — This value is the number of oscillator clocks to count between each refresh. For example, with a 50MHz oscillator, programming the value to 02EEh will result in a 15µsec refresh rate. Programming the register to 1770h with the same oscillator will result in a 120µsec refresh rate.
Refresh period = (Oscillator period) X (Refresh Rate Register Value)
If the oscillator clock runs at 50 MHz and therefore has a 20 nsec period, and the Refresh Rate Register is set to 02EEh (750):
(20nsec) X (750) = 15 µsec.

3.3.1.5.3 RAS Timeout Register

This 16-bit register is used to set how long the $\overline{RAS}$ pins may be kept low. If the $\overline{RAS}$ low time exceeds the value programmed into this register, the DRAM Controller will force that $\overline{RAS}$ signal high, resulting in a page miss on the next access. If the $\overline{RAS}$ low time is greater than the refresh period, then this register value can be programmed to a value greater than the Refresh Rate Register, and the counter will have no effect. At reset, this register is initialized to: 1388h. The RAS Timeout Register is written through I/O address EF84h.



I/O Map Address	Access
EF84h	R/W

Bits 15-13: Reserved

Bits 12-0: RAS Rate Divisor — This value is the maximum number of oscillator clocks $\overline{RAS}$ may remain low. If this count is reached, the DRAM controller will force $\overline{RAS}$ high, resulting in a page miss. The count is reset when $\overline{RAS}$ goes high due to a refresh or a page miss.
RAS Timeout Period = (Oscillator period) X (RAS Timeout Register value + 1)
If the oscillator clock runs at 50 MHz and therefore has a 20 nsec period, and the RAS Timeout Register is loaded with

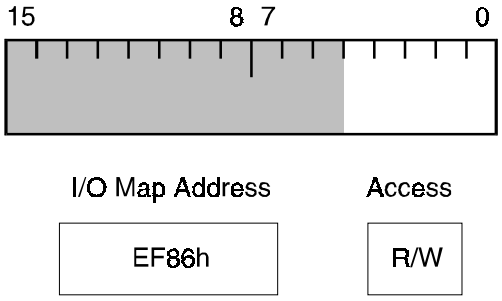
01770h (6000 decimal):

(20 nsec) X (6000) = 120 µsec.

Note: If the RAS Timeout value is changed during operation, the new value will not take affect until after the next refresh period.

3.3.1.5.4 DRAM Status Register

This 16-bit register shows the status of the DRAM Controller. The DRAM Status Register is written through I/O address EF86h.



Bits 15-5: Reserved

Bit 4: Bank 1 Parity Error (read only)
0 = No error in Bank 1 (default)
1 = Parity error in Bank 1

Bit 3: Bank 0 Parity Error (read only)
0 = No error in Bank 0 (default)
1 = Parity error in Bank 0

Bit 2: Clear Parity Error (write only, self-clearing)
0 = No effect
1 = Clear Parity Error. Will clear itself back to 0.

Bit 1: Parity Error Detected (read only)
0 = No Parity error (default)
1 = Parity error detected

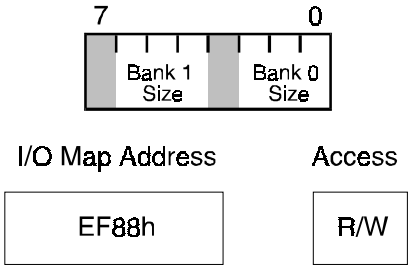
Bit 0: Initialization Complete (read only)
0 = DRAM Disabled/Initialization incomplete (default)
1 = Initialization complete

During initialization, Bit 0 can be checked to determine when the DRAM controller is ready for operation.

When a parity error causes an NMI, the interrupt service routine can quickly check if it was caused by a DRAM parity error by reading Bit 1. If it is set to one, Bits 3 and Bits 4 can be checked to determine which bank had the parity error. Finally, writing a 1 to Bit 2 clears the parity error bits, Bits 1, 3, and 4.

3.3.1.5.5 DRAM Bank Size Register

This 8-bit register is used to select the size of each bank of DRAM. The DRAM Bank Size Register is written through I/O location EF88h.



Bit 7: Reserved

Bits 6-4: DRAM Bank 1 Size —Each bank of memory is 16-bits wide, so, for example, a 1Mbyte bank size would probably consist of two 512Kx8/9 DRAMs.

Bit 6	Bit 5	Bit 4	DRAM Bank 1 Size (Mbytes)
0	0	0	0.5
0	0	1	1
0	1	0	2
0	1	1	4
1	0	0	8 (default)
1	0	1	Reserved
1	1	0	Reserved
1	1	1	Reserved

Bit 3: Reserved
Bits 2-0: DRAM Bank 0 Size

Bit 2	Bit 1	Bit 0	DRAM Bank 0 Size (Mbytes)
0	0	0	0.5
0	0	1	1
0	1	0	2
0	1	1	4
1	0	0	8 (default)
1	0	1	Reserved
1	1	0	Reserved
1	1	1	Reserved

3.3.1.5.6 Bank Addressing and Masks

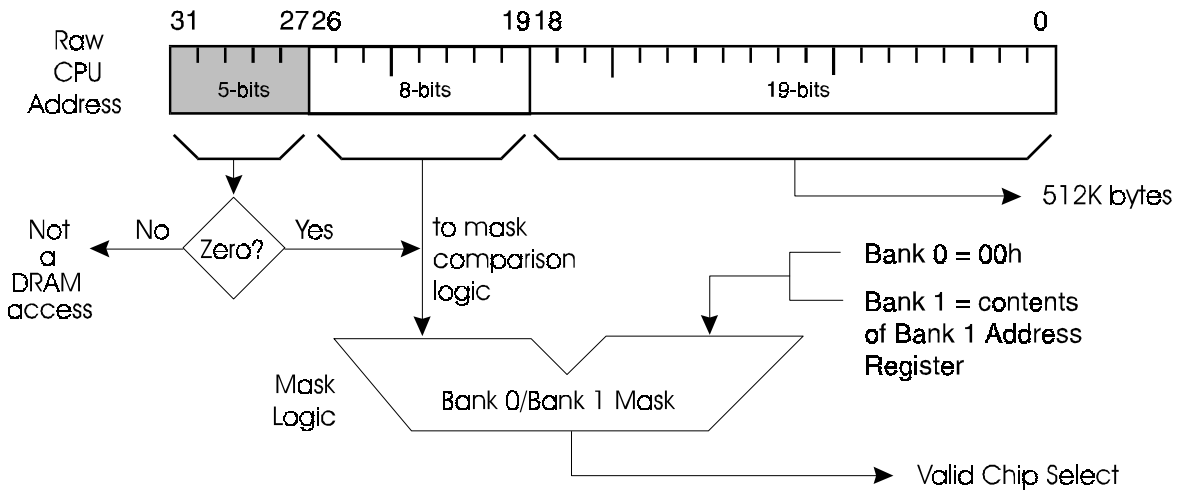
To determine if a CPU local bus cycle is to access DRAM, the DRAM controller decodes the address as well as the cycle type. The DRAM controller's address comparison logic compares the CPU local bus address with the programmed address range(s) of the two memory banks. The user sets a valid range through a Bank 1 Address Register and two bank "mask" registers. The mask register value is compared to CPU local bus address bits A26-A19, and if there is match, the address is determined to be valid.

The address coming from the CPU can be broken into three groups of bits: 31-27, 26-19, and 18-0. For both banks, address bits 31-27 will always be zero when accessing the DRAM. For both banks, 18-0 determine the decoding for the lower 512K bytes. Bits 26-19 are compared with the values in the bank 1 address register and the two mask registers and a "true" comparison enables a chip select signal and the DRAM access.

When the DRAM controller is enabled, a CPU local bus access must meet the following three requirements to access DRAM Bank 0:

1 -- It must be a memory cycle.
2 -- The CPU local bus A31-A27 signals must all be zeroes.
3 -- Each of the CPU local bus A26-A19 signals must be a zero, or the corresponding bit in the Bank 0 Mask Register must be a zero.

Figure 3-3 Valid DRAM Address Chip Select Generation



When the DRAM controller is enabled and DRAM Bank 1 is also active, a CPU local bus access must meet the following three requirements to access DRAM Bank 1:

1 -- It must be a memory cycle.

2 -- The CPU local bus A31-A27 signals must all be zeroes.

3 -- Each of the CPU local bus A26-A19 signals must match the corresponding bit in the Bank 1 Address Register or the corresponding bit in the Bank 1 Mask Register must be a zero.

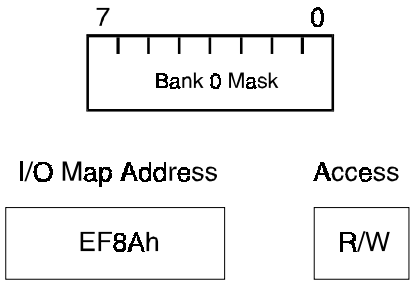
Typical addresses and masks are shown below:

Bank# (Start Address)	Bank 0 (0M)	Bank 1 (1M)	Bank 1 (8M)
A26-A19	00h	02h	10h
Mask Value	FFh	FEh	F0h

If a large DRAM device is used, the lower bits of the Mask Register should be set to zero. For example, a 1Mbyte DRAM device will use A19 as part of the direct address. Therefore, the Mask Register bit 0 should be set to 0 to allow this bit to be either 1 or 0. This allows a valid Chip Select throughout the entire 1 Mbyte address range.

3.3.1.5.7 Bank 0 Mask Register

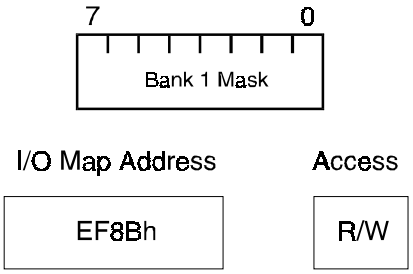
This 8-bit register is used to mask address bits 26-19 in the internal DRAM Controller chip select logic for Bank 0.



Bits 7-0: Bank 0 Mask — These eight bits mask the DRAM Controller Bank 0 chip select address comparison bits 26-19 which are always 00h in the case of Bank 0. Every bit set to a one in these registers indicates the associated chip select address (always 0) must match the CPU address for an active internal chip select to be decoded. If a register bit is programmed to a zero, then the associated chip select address comparison bit does not need to match. The upper 5 address bits (bits 31-27) must always be equal to 0.

3.3.1.5.8 Bank 1 Mask Register

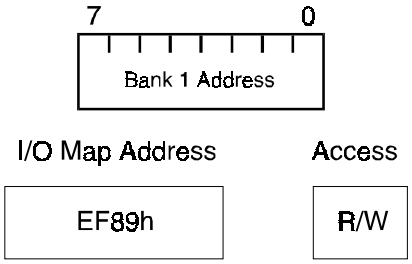
This 8-bit register is used to mask address bits 26-19 in the internal DRAM Controller chip select logic for Bank 1. It sets the high order location of Bank 1 memory in the memory map.



Bits 7-0: Bank 1 Mask — These eight bits mask the DRAM Controller Bank 1 chip select address comparison bits 26-19 as set in the Bank 1 Address Register. Every bit set to a one in these registers indicates the associated Bank 1 Address Register bit must match the CPU address for an active internal chip select to be decoded. If a register bit is programmed to a zero, then the associated chip select address comparison bit does not need to match.

3.3.1.5.9 Bank 1 Address Register

This 8-bit register is compared with address bits 26-19 in the internal DRAM Controller chip select logic for Bank 1.



Bits 7-0: Bank 1 Address — These eight bits must exactly match address bits 26-19 (unless masked in the Bank 1 Mask Register) for an active internal DRAM chip select for Bank 1 to be generated. The upper 5 address bits (bits 31-27) must always be equal to 0.

3.3.1.5.10 DRAM Timing Diagrams

Below are some typical read and write cycle timing diagrams. They show the relationship between clock cycles and valid data writes and reads.

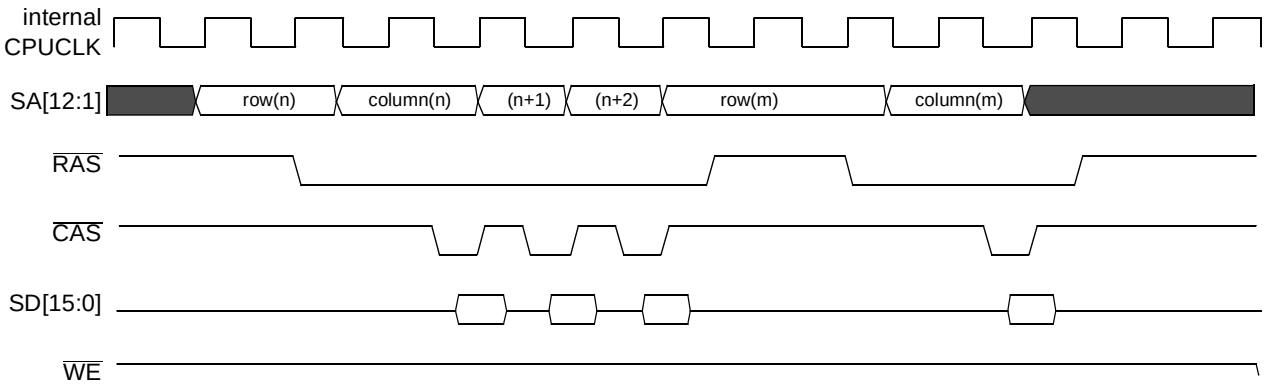


Figure 3-4 DRAM Read Cycle (4 cycle page miss)

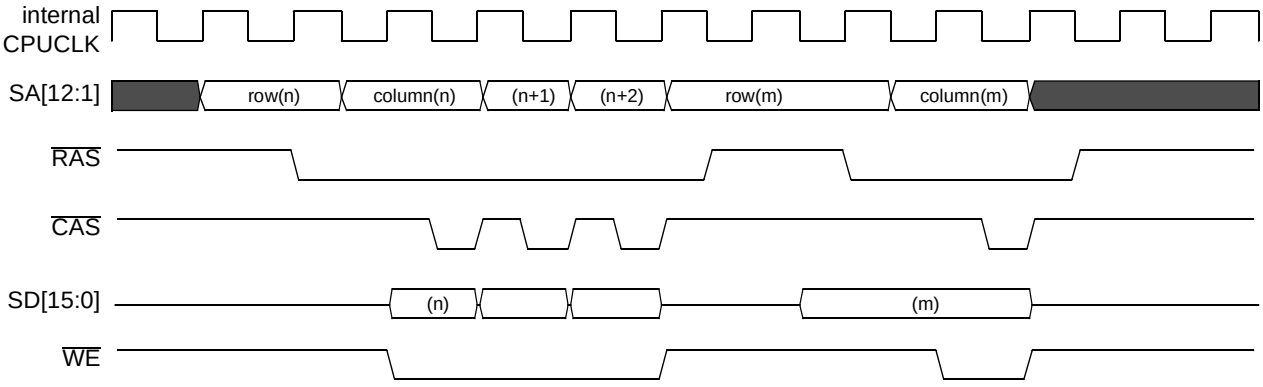


Figure 3-5 DRAM Write Cycle (4 cycle page miss)

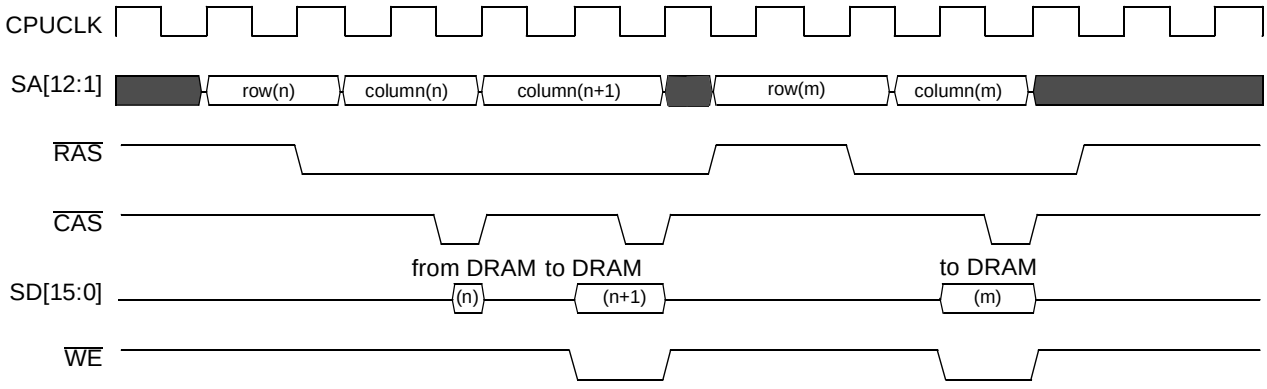


Figure 3-6 DRAM Read Followed by DRAM Write (4 cycle page miss)

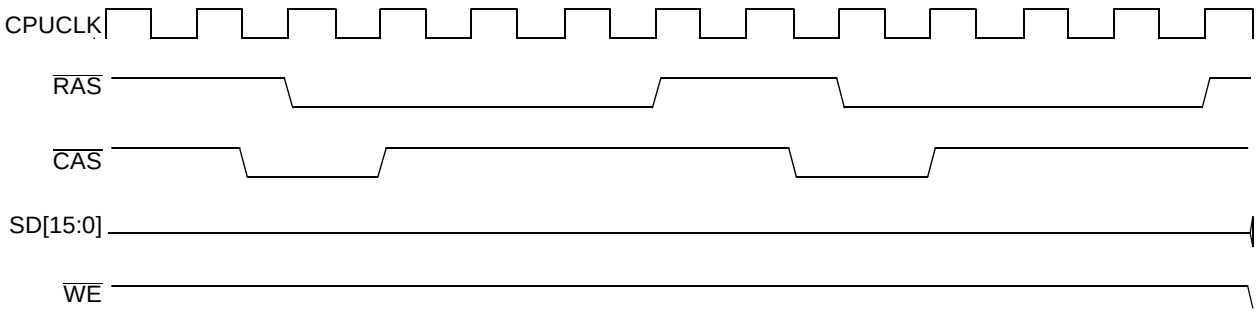


Figure 3-7 CAS Before RAS Refresh Cycle (4 cycle page miss)

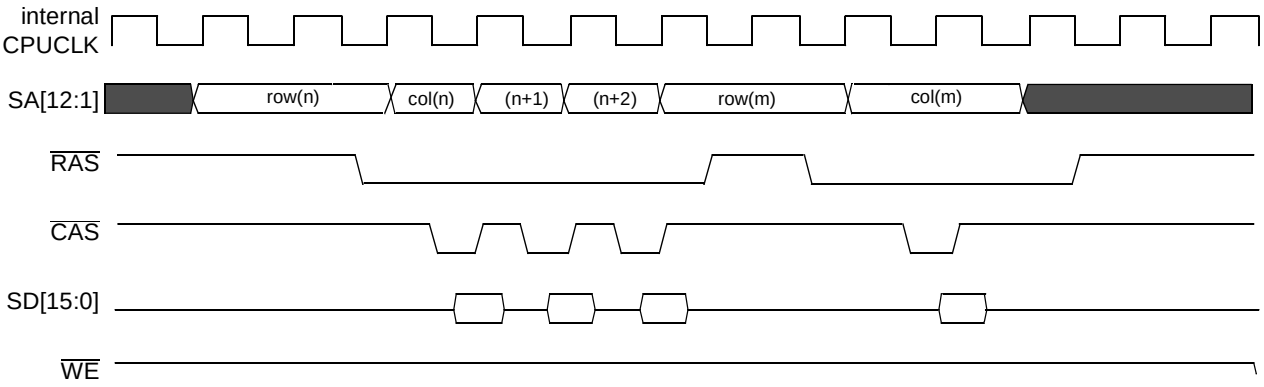


Figure 3-8 DRAM Read Cycle (3 cycle page miss)

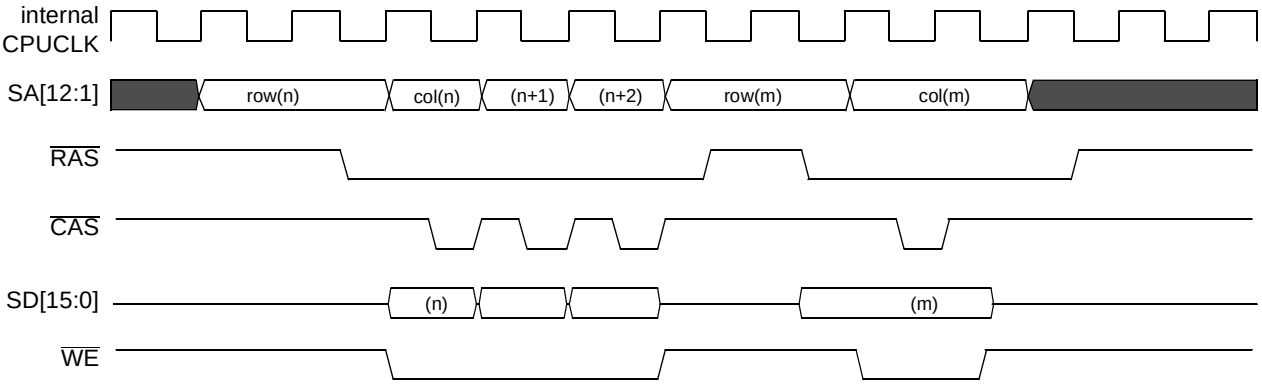


Figure 3-9 DRAM Write Cycle (3 cycle page miss)

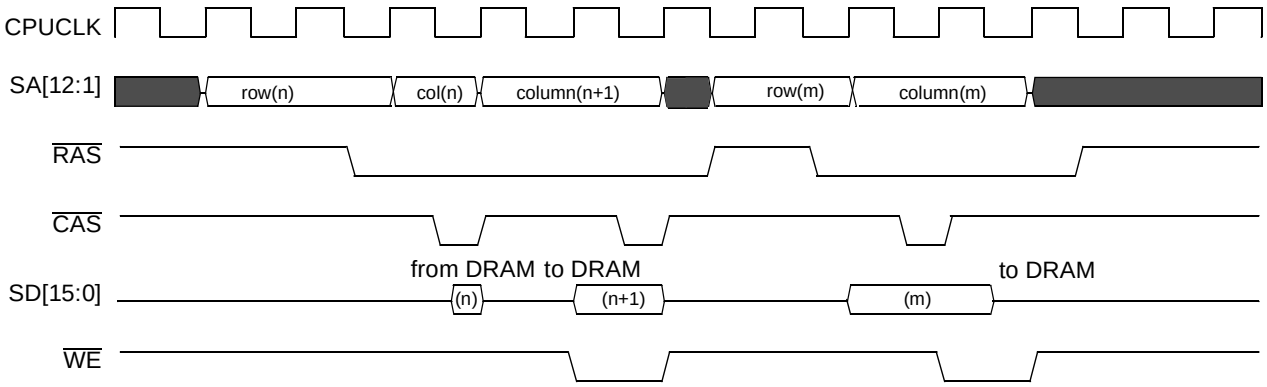


Figure 3-10 DRAM Read Followed by DRAM Write (3 cycle page miss)

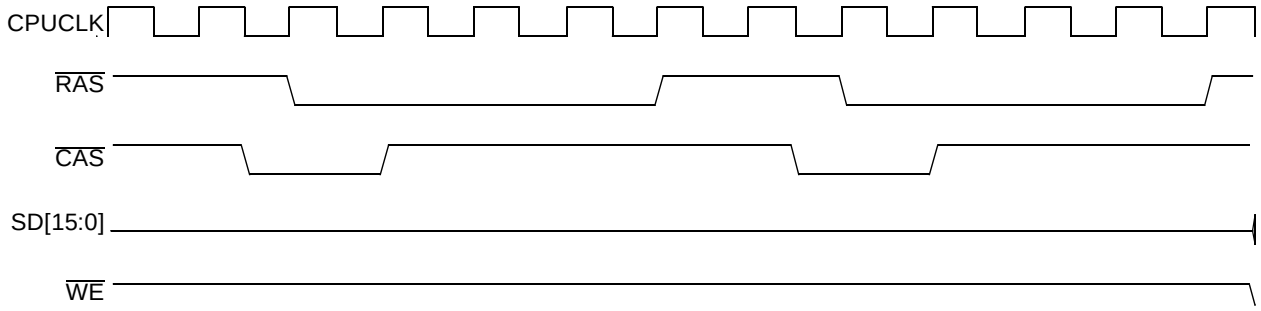


Figure 3-11 $\overline{\text{CAS}}$ Before $\overline{\text{RAS}}$ Refresh Cycle

3.3.2 The Programmable Interval Timer (PIT)

The NS486SXL programmable interval timer is compatible with the Intel 8254 programmable interval timer and contains three identical timers, CH0, CH1, and CH2. CH0 and CH1 can be used to generate accurate timing delays under software control. CH2 can be used to provide a WATCHDOG timer function.

The PIT is programmed through I/O ports 0040h-0043h. Three timer channels, CH0, CH1, and CH2, are outputs of three 16-bit presettable down counters that can be programmed to count in binary or binary coded decimal (BCD). The counters are completely independent and can operate either as timers or counters. Both can be programmed to operate in various modes. CH0 and CH1 are driven from either an external source or an internally generated clock signal divided by 4, 8, 16 or 32. The internally generated clock can be selected to be the CPU operating frequency or the raw input (oscillator) clock divided by two (See Section 3.3.7.) CH2 is driven by a 1 kHz clock provided by the Real Time Clock.

Note: To use the WATCHDOG timer function (CH2), the Real Time Clock's crystal circuit must be installed and operating.

Logic common to all of the counters reads and writes data to and from the 8-bit data port to the appropriate counter.

Each counter is identical and contains a Control Word Register, a Status Register, a 16-bit down counting element, two input holding registers, two output holding registers, a clock input, a gate input (GATE) and an out signal (OUT).

The PIT contains one register, the Control Word Register, which determines how the counters operate. When latched, the Status Register contains the current contents of the Control Word Register and the status of the output and null count flag. The 16-bit down counting element is presettable and synchronous.

The Input Holding Registers are two 8-bit latches used to store the LSB and MSB of the 16-bit count written to the counter until they are transferred to the counting element on the next falling edge of the appropriate clock input. The Output Holding Registers are two 8-bit latches used to latch the LSB and MSB of the present 16-bit count, thus enabling the count to be read.

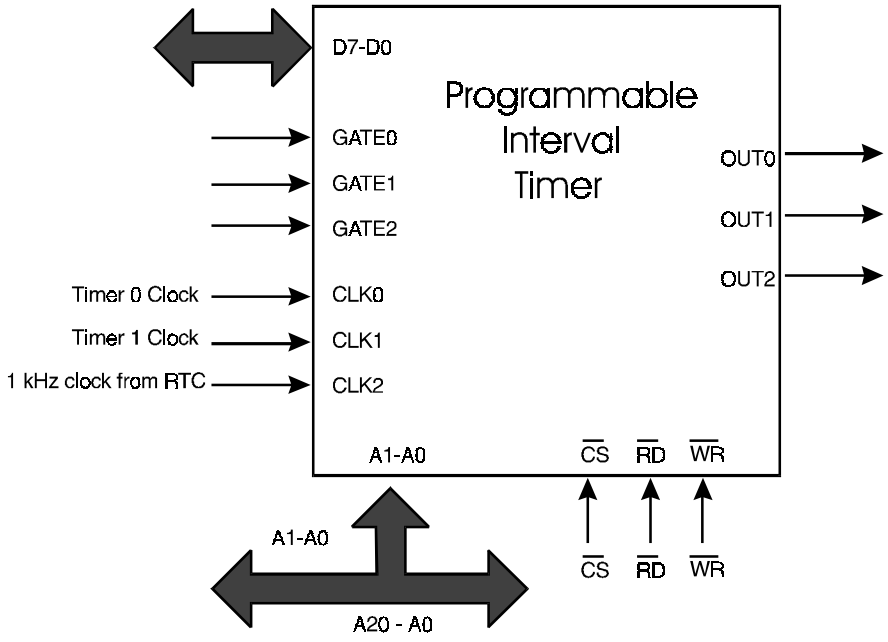


Figure 3-12 Programmable Interval Timer

The internally divided Oscillator clock provides the clock signal for loading and decrementing the counting element. The functionality of CH0's and CH1's GATE and OUT signals are defined by the counter mode and the status of the counting element.

3.3.2.1 Programming the PIT System

Immediately after a system reset, the PIT is not accessible via its I/O mapped registers until access is enabled via the Bus Interface Unit (BIU) Control Register 1 and 2. To enable accesses to the PIT, a "1" must be written to bit 3 of BIU Control Register 1 (I/O address EF00h). Refer to the BIU section for more information.

The PIT's registers and counters power up with random contents. Therefore, each counter must be programmed before it can be used. Counters are programmed by first writing to the Control Word Register, followed by writing an initial count to the appropriate counter. The table below shows the PIT register's I/O addresses.

I/O Address	PIT Register	Type of Access Permitted
0040h	Counter 0	R/W
0041h	Counter 1	R/W
0042h	Counter 2	R/W
0043h	Control Word Register	W only

3.3.2.2 Control Word Register

The Control Word Register is at I/O address 0043h and is a write only register. The Control Word Register determines the counter to be programmed, the counter's mode of operation, the method by which the counter will be read and written, as well as whether the counter is binary or BCD. Note that the Control Word Register will need to be written once for each counter. When a Control Word is written for a counter, all of that counter's control logic is immediately reset and its OUT signal goes to a known initial state.

Bit 7: Select Counter Bit 1 (SC1)
Bit 6: Select Counter Bit 0 (SC0)

These two bits select the counter to be programmed or the Read Back Command as follows:

Bit 7	Bit 6	Counter/Command Selected
0	0	Counter 0
0	1	Counter 1
1	0	Counter 2
1	1	Read Back Command

Bit 5: Read/Write Bit1 (RW1)
Bit 4: Read/Write Bit0 (RW0)

These two bits select the method that the counter selected with SC1-SC0 will be read or written, or the Counter Latch Command for the selected counter.

Bit 5	Bit 4	Read/Write Method or Command
0	0	Counter Latch Command
0	1	R/W LSB of counter only
1	0	R/W MSB of counter only
1	1	R/W LSB first followed by MSB of counter

The following table summarizes the command and counter accesses as programmed with the Control Word Register Bits 7-4.

Bit 7	Bit 6	Bit 5	Bit 4	Command/Format
0	0	0	0	Counter Latch Command for Counter 0
0	0	0	1	Counter 0 LSB R/W only
0	0	1	0	Counter 0 MSB R/W only
0	0	1	1	Counter 0 LSB and MSB R/W
0	1	0	0	Counter Latch Command for Counter 1
0	1	0	1	Counter 1 LSB R/W only
0	1	1	0	Counter 1 MSB R/W only
0	1	1	1	Counter 1 LSB and MSB R/W
1	0	0	0	Counter Latch Command for Counter 2
1	0	0	1	Counter 2 LSB R/W only
1	0	1	0	Counter 2 MSB R/W only

Bit 7	Bit 6	Bit 5	Bit 4	Command/Format
1	0	1	1	Counter 2 LSB and MSB R/W
1	1	X	X	Read Back Command

Bit 3: Mode Bit 2 (M2)
Bit 2: Mode Bit 1 (M1)
Bit 1: Mode Bit 0 (M0)

These three bits determine the mode of operation for the selected counter as follows:

Bit 3	Bit 2	Bit 1	Counting Mode	Mode Description
0	0	0	Mode 0	Interrupt on Terminal Count
0	0	1	Mode 1	Hardware Retriggerable One Shot
X	1	0	Mode 2	Rate Generator
X	1	1	Mode 3	Square Wave Mode
1	0	0	Mode 4	Software Triggered Strobe
1	0	1	Mode 5	Hardware Triggered Strobe

Bit 0: Binary Coded Decimal Bit (BCD)
0 = Selected Counter is a 16-bit Binary Counter
1 = Selected Counter is a 4 Decade BCD Counter

3.3.2.3 Counter Write Operations

For each counter, the Control Word Register must be written before the initial count. The initial count must follow the counter read/write format as programmed in the Control Word Register.

New initial counts may be written into a counter any time without re-issuing a control word, as long as the existing format is observed. This will not affect the counter's programmed mode. The counters are 16-bit and are accessed through an 8-bit port. This means that writes to a counter can be performed in one of three ways:

1. LSB only
2. MSB only
3. LSB followed by MSB.

The method of access is defined by the Control Word format. When a Control Word is written, the Input Holding Registers are automatically set to zeros.

Input Holding Registers
Bits 15-0: Count Bits -- These bits define the count value loaded in the counter.

3.3.2.4 Counter Read Operations

It is usually desirable to read the value of a counter without disturbing the present count in process. The NS486SXL PIT is able to provide this function in two ways: the Counter Latch and the Read-Back Command.

A count value may also be directly read from a counter. When this method is used, the CLK input for the selected counter must not be enabled, to ensure that a stable count value is read. If the count value is changing at the same instant that the count is read, an invalid count value will be read. When a counter is read (Counter 0 - 0040h, Counter 1 - 0041h and Counter 2 - 0042h), the 16-bit count value must be read in accordance with the read/write sequence programmed through the Control Word Register.

Current Count Registers
Bits 15-0: Count Bits -- These bits define the count value currently in the counter.

3.3.2.5 Counter Latch Command

This command is written to the Control Word Register (I/O address 0043H). The bits SC1 and SC0 select one of the three counters. Bits RW1 and RW0 distinguish this command from a Control Word. When the Counter Latch Command is received by the Control Word Register, the PIT's Output Holding Registers latch the count of the selected counter. This count is held until read by the CPU or the counter is reprogrammed.

If a counter's value is latched, and the Counter Latch Command is issued again before the CPU has read it, the second counter latch command will be ignored and the value in the Output Holding Register will be that of the first count latched. Similarly, issuing a Read Back command before the CPU has read the latched value will not update the latched count either. The latched count value must be read according to the previously programmed format in the Control Word Register.

Once the counter latch command has been issued for a counter, the value latched is read through the counter I/O port (Counter 0- 0040h, Counter 1 - 0041h, and Counter 2 - 0042h), according to the read/write format programmed in the control word.

Counter Latch Command (0043h) Bits:

Bit 7: Select Counter Bit1 (SC1)

Bit 6: Select Counter Bit0 (SC0)
These two bits select the counter to be latched as follows:

Bit 7	Bit 6	Counter Selected
0	0	Counter 0 (I/O Port 0040h)
0	1	Counter 1 (I/O Port 0041h)
1	0	Counter 2 (I/O Port 0042h)
1	1	Reserved

Bit 5: Read/Write Bit1 (RW1)

Bit 4: Read/Write Bit0 (RW0)
These two bits must equal 0 to issue the Counter Latch Command.

Bits 3-0: Don't Care for the Counter Latch Command

Output Holding Register

Bits 15-0: Count Bits
These bits define the count value latched with the Counter Latch Command.

3.3.2.6 Read-Back Command

This command is written to the Control Word Register (address 0043H). The bits SC1 and SC0 distinguish this command from a Control Word. When these bits are 1,1 the Control Word becomes the Read-Back command and provides the following alternative bit definitions:

Bit 7: Select Counter Bit1 (SC1)

Bit 6: Select Counter Bit0 (SC0)
These two bits must be equal to 1 to issue the Read-Back Command.

Bit 5: Count Bit (COUNT)
0 = Latch Count of the Selected Counters
1 = Do Not Latch Count of the Selected Counters

Bit 4: Status Bit (STATUS)
0 = Latch Status of the Selected Counters
1 = Do Not Latch Status of the Selected Counters

Bit 3: Count 2 Bit (CNT2)
0 = Do not select Counter 2
1 = Select Counter 2

Bit 2: Count 1 Bit (CNT1)
0 = Do not select Counter 1
1 = Select Counter 1

Bit 1: Count 0 Bit (CNT0)
0 = Do not select Counter 0
1 = Select Counter 0

Bit 0: Reserved: 0

The Read-Back Command allows the user to check the value of a selected counter, determine its programmed mode and monitor the status of the OUT signal and Null Count Flag.

This command may be used to latch multiple Output Holding Registers for the counters in the PIT. By setting Bit 5 to zero and selecting the desired counters to be read, the counters' Output Holding Registers can be latched at the same instant, thus enabling all counters to be latched at the same instant. When reading the counters, the pre-programmed format should be observed. The specific counter is automatically un-

latched when read or when the counter is reprogrammed.

Output Holding Registers:

Bits 15-0: Count Bits — These bits define the count value latched with the Read-Back Command.

The Read-Back Command can also latch status information of selected counters by setting Bit 4 to zero. The Status Register must be latched to be read. The status of a counter is obtained by a read from the selected counter when the Status Register is latched.

This is the Status Byte:

Bit 7: Output Bit
0 = signal is 0
1 = signal is 1

Bit 6: Null Count Bit
0 = Count can be read (Count has been loaded)
1 = Null Count (Count has not been loaded)
This bit indicates if the count has been loaded into the counter. The instant this happens is Mode dependent.

Bit 5: Read/Write Bit1 (RW1)

Bit 4: Read/Write Bit0 (RW0)
These bits reflect the counter's programmed read/write format as follows:

Bit 5	Bit 4	Programmed Read/Write Method
0	0	Counter Latch Command
0	1	R/W LSB of counter only
1	0	R/W MSB of counter only
1	1	R/W LSB first followed by MSB of counter

Bit 3: Mode Bit2 (M2)

Bit 2: Mode Bit1 (M1)

Bit 1: Mode Bit 0 (M0)
These three bits reflect the counter's programmed mode as follows:

(Section 3.3.2.10 on page 45 for a complete description of each mode)

Bit 3	Bit 2	Bit 1	Counting Mode	Mode Description
0	0	0	Mode 0	Interrupt on Terminal Count
0	0	1	Mode 1	Hardware Retriggerable One Shot
X	1	0	Mode 2	Rate Generator
X	1	1	Mode 3	Square Wave Mode
1	0	0	Mode 4	Software Triggered Strobe
1	0	1	Mode 5	Hardware Triggered Strobe

Bit 0: Binary Coded Decimal Bit (BCD)
0 = Counter is programmed to be a 16-bit Binary Counter
1 = Counter is programmed to be a 4 Decade BCD Counter

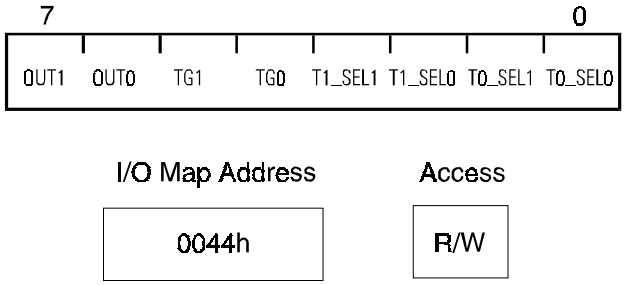
Both the Status and the Count of the selected counters can be latched simultaneously by setting bits 4 and 5 to zeros. The first read after this will always return the status byte. The following one or two reads return the latched count depending on the programmed read/write mode for the counter. Subsequent reads return unlatched counts.

3.3.2.7 External Timer Control Signals

The function of the T1 and T0 pins of the NS486SXL are determined by bits 3-0 of this register. The combinations supported provide the user with access to two function signals on either Timer 0 or Timer 1; or access to one function signal on Timer 0 and one on Timer 1.

This register is located at I/O address 0044h and has a reset value of 00h.

3.3.2.8 Timer I/O Control Register



Bit 7: OUT1 — Timer 1 OUT signal. When read, this bit is the state of Timer 1's OUT signal. This bit is not writable.

Bit 6: OUT0 — Timer 0 OUT signal. When read, the state of Timer 0's OUT signal will be read. This bit is not writable.

Bit 5: TG1 — Timer 1 GATE. When neither the T1 pin nor the T0 pin is selected to drive the GATE input to Timer 1, this bit determines the state of the GATE signal associated with Timer 1.

Bit 4: TG0 — Timer 0 GATE. When neither the T1 pin nor the T0 pin is selected to drive the GATE input to Timer 0, this bit determines the state of the GATE signal associated with Timer 0.

Bits 3-2: T1_SEL1, T1_SEL0 — T1 pin function selection bits 1 and 0. These two bits determine the function of the T1 pin of the NS486SXL. If these bits and bits 1-0 of this register result in both T1 and T0 driving either GATE0 or GATE1, then these two signals will be logically ORed to make a single GATE# signal.

T1_SEL1	T1_SEL0	Function of T1 pin
0	0	Timer 0's GATE input signal (GATE0).
0	1	Timer 1's GATE input signal (GATE1).
1	0	Timer 1's OUT output signal (OUT1).
1	1	Timer 0's CLK input signal (CLK0)

Note: When the T1 pin does not drive Timer 0's CLK input, the programmable timer clock controlled by the Timer Clock Register will drive the CLK0 input.
When switching between these two possible CLK0 sources, CLK0 may glitch, so the user should not use the timer until after first selecting the appropriate clock source.

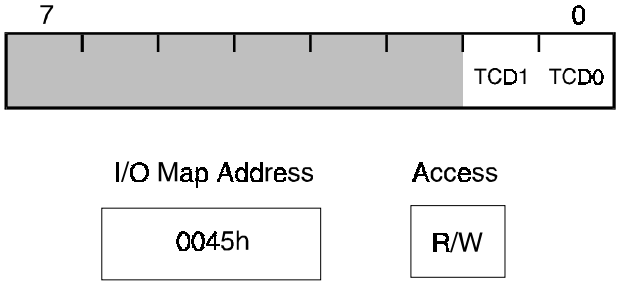
Bits 1-0: T0_SEL1, T0_SEL0 — T0 pin function selection bits 1 and 0. These two bits determine the function of the T0 pin of the NS486SXL. If these bits and bits 3-2 of this register result in both T1 and T0 driving either GATE0 or GATE1, then these two signals will be logically ORed together to make a single GATE# signal.

T0_SEL1	T0_SEL0	Function of T0 pin
0	0	Timer 0's GATE input signal (GATE0).
0	1	Timer 1's GATE input signal (GATE1).
1	0	Timer 0's OUT output signal (OUT0).
1	1	Timer 1's CLK input signal (CLK1).

Note: When the T0 pin does not drive Timer 1's CLK input, the programmable timer clock controlled by the Timer Clock Register will drive the CLK1 input.
When switching between these two possible CLK1 sources, CLK1 may glitch, so the user should not use the timer until after first selecting the appropriate clock source.

3.3.2.9 Timer Clock Register

This register produces a clock signal which may be used by Timer 0 or Timer 1. This programmable clock divides the frequency of a selected internal clock by 4, 8, 16 or 32. The clock selected may be either the CPU operating clock (affected by Power Management Power Save Modes) or the "raw" oscillator clock divided by two. This register should be programmed before the clock it produces is used. This register is located at IO address 0045h and has a reset value of 00h. The input clock frequency is selected by programming Power Management Register 3. See Section 3.3.7.5.3.



Bits 7-2 Reserved.

Bits 1-0: TCD1, TCD0 — Timer Clock Divisor bits 1 and 0. These two bits determine the divisor selected to generate a clock which may be used by Timer 1 and/or Timer 0.

TCD1	TCD0	Resulting Clock
0	0	Selected clock divided by 4.
0	1	Selected clock divided by 8.
1	0	Selected clock divided by 16.
1	1	Selected clock divided by 32.

Whenever, the T1 pin is not selected to drive Timer 0's input CLK signal (see bits 3-2 of the Timer I/O Control Register), this clock source will provide the clock for Timer 0.

Whenever, the T0 pin is not selected to drive Timer 1's input CLK signal (see bits 1-0 of the Timer I/O Control Register), this clock source will provide the clock for Timer 1.

3.3.2.10 Mode Descriptions

In the following Mode Descriptions section the terms GATE and OUT are used generically to apply as each timers GATE input signal and OUT output signal.

3.3.2.10.1 Mode 0: Interrupt on Terminal Count

After the Control Word is written, OUT is low and remains low until the count reaches zero. Then OUT goes high and remains high until a new count or a new Control Word is written. GATE = 1 enables counting, GATE = 0 disables counting.

The first clock pulse after a Control Word and initial count has been written loads initial count. This clock pulse does not decrement the count. If a two byte count is written, the first byte disables counting and OUT is set low. The second byte allows the new count to be loaded on the next clock pulse. An initial count can be written when GATE = 0 and will still be loaded on the next clock pulse. No clock pulse is needed to load the initial count when GATE goes to a 1.

3.3.2.10.2 Mode 1: Hardware Retriggerable One-Shot

Initially high OUT will go low on the clock pulse after a trigger. This begins the one-shot pulse. OUT remains low until the count reaches zero. OUT then goes high and remains high until the next trigger, then goes low on the next clock pulse. After writing a Control Word and initial count the counter is armed. A trigger loads the initial count and sets OUT low on the next clock pulse. The one-shot is Re-triggerable and can be repeated without writing the same count or Control Word into a counter. GATE has no effect on OUT.

3.3.2.10.3 Mode 2: Rate Generator

OUT is initially high. When the initial count is decremented to 1, OUT goes low for one clock pulse, then goes high again. The counter reloads the initial count and the process is repeated. This same sequence is repeated indefinitely. GATE = 1 enables counting, GATE = 0 disables counting. If GATE goes low during an output pulse, OUT goes high immediately. A trigger loads the counter with the initial count on the next clock pulse. After the trigger, OUT goes low when the count has expired. After writing a Control Word and initial count, the counter will be loaded on the next clock pulse. After the initial count is written, OUT goes low when the count has expired, thus the synchronizing of the counter by software can be achieved. Writing a new count while counting does not affect the current counting sequence. If a trigger is received after writing a new count, but before the end of the current period, the counter will be loaded with the new count on the next clock pulse. Counting will continue from the new count. A count of 1 is illegal in Mode 2.

3.3.2.10.4 Mode 3: Square Wave

Typically used for Baudrate generation, OUT will be initially high, and goes low when half of the initial count has expired and remains low for the remainder of the count. This is periodic and the sequence is continued indefinitely. This is similar to Mode 2. GATE = 1 enables counting, GATE = 0 disables counting. If GATE goes low during an output pulse, OUT goes high immediately. A trigger reloads the counter with the initial count on the next clock pulse. After the trigger, OUT goes low when the count has expired. After writing a Control Word and initial count, the counter will be loaded on the next clock pulse. After the initial count is written, OUT goes low when the count has expired, thus the synchronizing of the counter by software can be achieved. Writing a new count while counting does not affect the current counting sequence. If a trigger is received after writing a new count but before the end of the current half-cycle of the square wave, the counter will be loaded with the new count on the next clock pulse. Counting will continue from the new count. Otherwise, the new count will be loaded at the end of the half-cycle.

For even counts, OUT is initially high. The initial count is loaded on the next clock pulse and is decremented by two on succeeding clock pulses. When the count expires, OUT goes low. The count is reloaded and the process repeats indefinitely.

For odd counts, OUT is initially high. The initial count minus one is loaded on the next clock pulse and is decremented by two on succeeding clock pulses. One pulse after the count expires, OUT goes low. The count is reloaded with the initial count minus one and the process repeats indefinitely.

3.3.2.10.5 Mode 4: Software Triggered Strobe

With OUT initially high, OUT goes low for one clock pulse and then goes high when the initial count expires. Writing the initial count triggers the counter. GATE = 1 enables counting, GATE = 0 disables counting. GATE has no effect on OUT. The counter will be loaded on the next clock pulse after writing the Control Word and initial count. This clock pulse does not decrement the count. A new count, written during counting, will be loaded on the next clock pulse and counting will continue from the new count. If a two byte count is written, the first byte will have no effect on counting, and the second byte will allow the new count to be loaded on the next clock pulse.

3.3.2.10.6 Mode 5: Hardware Triggered Strobe

OUT is initially high. Counting is triggered by a rising edge on the GATE input. OUT will go low for one clock pulse when the count has expired. After writing the initial count and Control Word, the counter will not be loaded until the clock pulse after the trigger. This pulse does not decrement the count. The counter is loaded on the next clock pulse after a trigger. GATE has no effect on OUT. If a new count is written during counting, the current count will not be affected. If a trigger is issued before a count expires, but after a new count is written, the counter will be loaded with the new count on the next clock pulse and counting will continue from there.

3.3.2.11 Gate Input

Sampling of the GATE input occurs on the rising edge of the CLK input to the counter. The GATE input is level sensitive and the logic level is samples on the rising edge of the clock in modes 0, 2, 3 and 4. In modes 1, 2, 3 and 5 the GATE input is rising edge sensitive. In these latter modes, a rising edge of the GATE sets an edge sensitive flip-flop in the counter. This flip-flop is sampled on the next rising edge of the clock. Immediately after the flip-flop is sampled, it is reset. This ensures a trigger (rising edge of the GATE input) will always be detected.

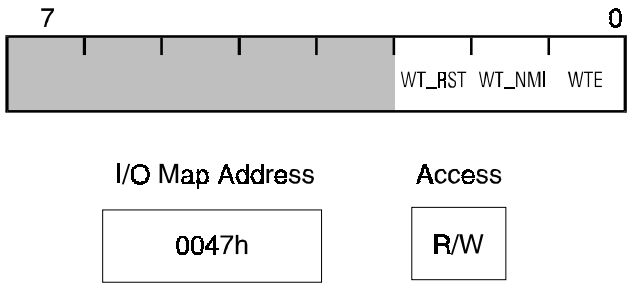
3.3.3 The WATCHDOG Timer

The NS486SXL WATCHDOG timer, CH2, is a protected 16-bit timer that can be used to prevents system "lockups or hangups". It uses a 1 KHz clock generated by the on-chip Real Time Clock circuit.

When the WATCHDOG timer is enabled it must be reset by the CPU before it times out. If the WATCHDOG Timer is enabled and times out, either a reset or a non-maskable interrupt (NMI) will be generated, based upon the values in bits 1 and 2 of the WATCHDOG Timer Control Register.

3.3.3.1 WATCHDOG Timer Control Register

The WATCHDOG Timer Control Register may be used to enable the WATCHDOG Timer and select the action to take when the WATCHDOG Timer times out. This register is located at I/O address 0047h and all of its bits will be reset to zero by a hardware reset.



Bits 7-3: Reserved

Bit 2: WT_RST — WATCHDOG Timer ReSeT. When the enabled WATCHDOG Timer times out and this bit is a one, the entire NS486SXL will be reset except for this bit. A hardware reset (i.e. PWGOOD going low) will reset this bit to a zero.
This bit may only be written when the WATCHDOG Timer is disabled (i.e. WTE = 0). This bit may be written at the same time the WTE bit is written to a one.

Bit 1: WT_NMI — WATCHDOG Timer Non-Maskable Interrupt. The first time the enabled WATCHDOG Timer times out, WT_RST is zero and this bit is a one, a NMI will be generated to the CPU. If the WATCHDOG Timer times out a second time under the above conditions without having been reset after the first CPU NMI, the entire NS486SXL will be reset except for this bit. A hardware reset or setting WT_RST to a one will reset this bit to a zero.
This bit may only be written when the WATCHDOG Timer is disabled (i.e. WTE = 0). This bit may be written at the same time the WTE bit is written to a one.

Bit 0: WTE — WATCHDOG Timer Enable. This bit enables/disables the WATCHDOG Timer function. When this bit is set to a one, all accesses to the Timer 2 registers in the programmable interval timer will be ignored

When this bit has been set, writes attempting to modify this register will also be ignored. The only ways to reset this bit back to a zero are:

1) Hardware Reset (PWGOOD goes low).
2) A WATCHDOG Timer generated reset.
3) An I/O write to address 0047h with data 69h, followed by an I/O write to address 0046h with data 5Ah.

3.3.3.2 Retriggering the WATCHDOG Timer Count to Prevent Resets

To reset the WATCHDOG Timer's count, the 16-bit value 8421h must be written to IO address 0046h. This will restart the WATCHDOG Timer and prevent either a reset or an NMI.

3.3.3.3 Interrupt Request Supported

The WATCHDOG Timer may also be configured via the Internal Interrupt Steering logic to drive a number of possible interrupt request lines when it times out. In this manner, the user may write zeros to both bits 1 and 2 of the WATCHDOG Timer Control Register and use a standard interrupt request to request servicing. Similar to the NMI functionality, if the WATCHDOG Timer times out twice before being reset, then the entire NS486SXL will be reset.

The user should be warned that using interrupt requests to service the WATCHDOG Timer is an unprotected method of operation. A run-away program may corrupt the interrupt request selection registers and effectively prevent the interrupt request from reaching the CPU. However, in such a case, ultimately the WATCHDOG Timer will timeout twice and reset the entire NS486SXL.

3.3.3.4 Programming Timer 2

To use Timer 2 as a WATCHDOG Timer, the user should program Timer 2 to operate in Mode 5: Hardware Triggered Strobe. The 16-bit count value may be programmed per the user's needs. The user should also note that Timer 2 may only be programmed when the WATCHDOG Timer is disabled (i.e. WTE = 0).

3.3.4 The MICROWIRE or Access.bus Interface

The **NS486SXL** MICROWIRE/Access.bus interface provides for full support of both the three wire MICROWIRE™ and the two wire Access.bus synchronous serial interfaces. These industry standard interfaces permit easy interfacing to a wide range of low-cost specialty memories and I/O devices. These include EEPROMs, SRAMs, timers, A/D converters, D/A converters, clock chips, and peripheral drivers.

The MICROWIRE or Access.bus interfaces share the use of two **NS486SXL** pins, so only one of them may be selected at any time. The two pins shared by the two interfaces are:

SCLK (pin #20), which both the MICROWIRE and Access.bus interfaces use as the serial clock.

SI (pin #19), which MICROWIRE uses as a serial input data pin, while Access.bus uses it as its I/O data pin.

The MICROWIRE interface requires a third pin:

SO (pin #18), which is used as a serial output data pin.

3.3.4.1 MICROWIRE Interface

MICROWIRE is a three-wire synchronous serial bus. All data transfers are synchronous with the serial clock (SCLK) and data is read into the **NS486SXL** MICROWIRE interface on the serial input pin (SI). Data written out will be driven on the serial output pin (SO).

Several MICROWIRE devices can be connect to the same three-wire system, as shown in Figure 3-13. One, and only one of these devices operates in what is called master mode and supplies the synchronous serial clock (SCLK) signal for the entire system. The master is also responsible for initiating all data transfers. All other MICROWIRE devices must operate in slave mode, where SCLK is an input signal and the slave device provides(receives) the serial data for read(write) cycles from(to) it. The slave device uses the master's SCLK for serially shifting data out(in), while the master device shifts the data in(out).

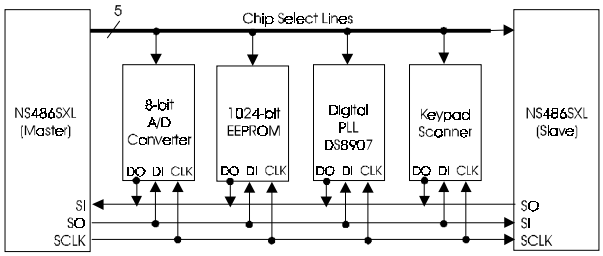


Figure 3-13 A MICROWIRE System Example

The **NS486SXL** MICROWIRE interface may be configured to operate in either master or slave mode. When the Configuration bit (CONFIG, bit 1 in the MICROWIRE Control Register) is a one, the **NS486SXL** MICROWIRE interface will be configured in master mode; when CONFIG is a zero, the **NS486SXL** MICROWIRE interface will be operate in slave mode.

When there is only a single MICROWIRE slave connected to the MICROWIRE master, the slave's chip select signal can be tied active since all MICROWIRE accesses should be to the slave device.

When more than one MICROWIRE slave is connected to the MICROWIRE master, each slave's chip select should be driven with a unique chip select signal. When configured as the MICROWIRE master, the **NS486SXL** MICROWIRE interface does not have any chip select logic designed into it. Instead it is suggested that the system designer configure the appropriate number of Reconfigurable I/O (RIO) pins to provide the necessary chip selects for the MICROWIRE slaves. These chip selects would then be under software control via these general purpose RIO pins and the software would be responsible for selecting the appropriate device for each MICROWIRE transfer. See Section 6.1 on page 107 for information on programming the RIO pins.

When the **NS486SXL** MICROWIRE interface is configured in slave mode, one of the RIO pins will operate as the MICROWIRE interface chip select input pin. The pin which will act as the input MICROWIRE chip select will be selected by the MICROWIRE Slave Chip Select Register. See "MICROWIRE Slave Chip Select Register" on page 59.

NOTE: In slave mode, the input MICROWIRE chip select is an active low signal, so the selected pin must be driven active low to select the **NS486SXL** MICROWIRE interface as a slave.

3.3.4.2 MICROWIRE Transfers

An 8-bit shift register, called Serial Input/Output (SIO) register, is used for both transmitting and receiving data. For either type of transfer, the bits of SIO are shifted left through the register. When the data byte is being transmitted, the bits are shifted out through the SO output pin (most significant bit first). When a data byte is being received, the bits are shifted in through the SI input pin (most significant bit first).

3.3.4.2.1 MICROWIRE Master Transmit/Receive

The **NS486SXL** MICROWIRE interface performs a transmit and receive at the same time. In master mode, the following steps will generate an 8-bit transfer:

1) Software must enable the MICROWIRE interface and configure it appropriately via the MICROWIRE Control Register.

2)If the **NS486SXL** MICROWIRE interface will be used to transmit data out, software must write the desired transmit data into the SIO register. If the **NS486SXL** MICROWIRE interface is only receiving data, one can ignore this step.

3) Software must set the BUSY bit in the MICROWIRE Control Register to a one. This will start the transfer.

4) At the end of the transfer, the MICROWIRE Interrupt Flag bit (uWI, bit 0 of the MICROWIRE Control Register) will be set to a one, an interrupt will be generated and the BUSY bit will be cleared to a zero. For receive transfers, the software should read the SIO register; this will clear both the interrupt request and the MICROWIRE Interrupt Flag bit. The other way to clear the interrupt request and uWI is to write a one to the BUSY bit; this is the typical method for multiple back to back Master transmit cycles.

5) If the next MICROWIRE transfer is a transmit go to Step 2. If the next MICROWIRE transfer is a receive, proceed to Step 3.

In master mode, whenever the BUSY bit is a zero, SCLK will remain low. Once the BUSY bit is set to a one, the MICROWIRE interface will proceed to generate eight periodic clock pulses on SCLK. For normal SCLK mode, on the rising edge of SCLK, the MICROWIRE interface will shift in the data presented on the SI pin and on the falling edge, the next bit in the SIO register will be shifted out onto the SO pin as depicted in Figure 3-14.

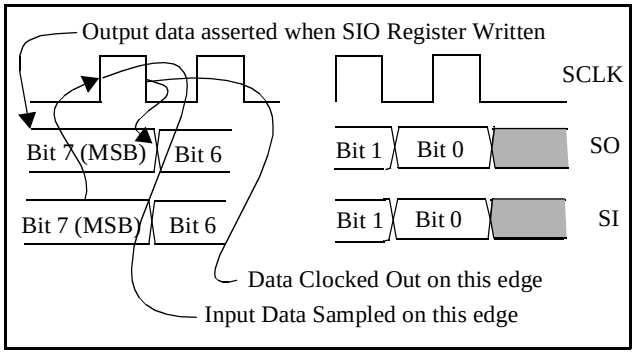


Figure 3-14 MICROWIRE Normal SCLK Mode Timing Diagram

The **NS486SXL** MICROWIRE interface also supports an alternate SCLK clocking mode. The timing of the serial transfer in alternate SCLK mode is slightly different. In alternate SCLK mode, the SIO register data being shifted out onto the SO pin occurs on the rising edge of SCLK. While the data being shifted in off of the SI pin, will be shifted in on the falling edge of SCLK. The timing of alternate SCLK mode is shown in Figure 3-15. The SCLK mode is determined

by the SCTL bit (bit 2 of the MICROWIRE Control Register).

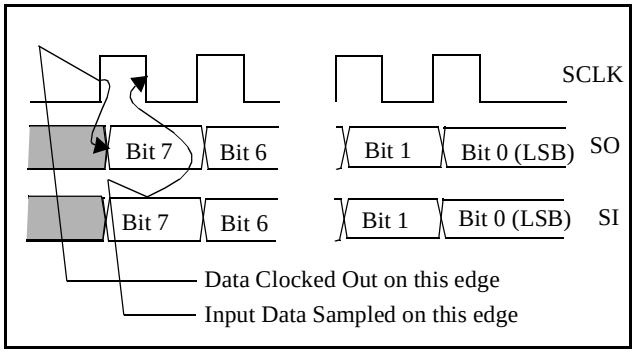


Figure 3-15 MICROWIRE Alternate SCLK Mode Timing Diagram

At the end of the transfer, the data shifted in on the SI pin will be readable via the SIO register and the BUSY bit will be cleared; thus SCLK will remain low until the BUSY bit is set to a one again. Also, the MICROWIRE transmit/receive interrupt flag (uWI, bit 0 of the MICROWIRE Control Register) will be set and an interrupt request will be generated.

3.3.4.2.2 MICROWIRE Slave Transmit/Receive

When the **NS486SXL** MICROWIRE interface is configured in slave mode, it must still perform the two steps required for a master transfers.

However, there are two major differences from master mode. First, the **NS486SXL** MICROWIRE interface will not respond unless its chip select pin is driven active low and second, the **NS486SXL** MICROWIRE interface will not generate the SCLK pulses. Instead, in slave mode the MICROWIRE interface must wait for the MICROWIRE master to generate these SCLK pulses and present (receive) the data appropriately.

At the end of a transmit cycle, the BUSY bit will be cleared to a zero, the uWI bit will be set and an interrupt generated. It is the responsibility of the software to guarantee the BUSY bit is set and the appropriate transmit data is written to the SIO register before the MICROWIRE master attempts to read from the **NS486SXL** MICROWIRE interface again. If the MICROWIRE master attempts another transfer before the BUSY bit is set, the **NS486SXL** MICROWIRE interface will not provide the appropriate data. This is an overall system software issue, which is beyond the scope of this document other than to note the issue.

At the end of the reception, the BUSY bit will be cleared to a zero, the uWI bit will be set and an interrupt generated. It is the responsibility of the software to guarantee that the SIO register is read and the BUSY bit is set before the MICROWIRE master attempts to transmit to the **NS486SXL** MICROWIRE interface again. If the MICROWIRE master attempts another transfer before the BUSY bit is set, the **NS486SXL** MICROWIRE interface will not clock in the appropriate data. User software should make sure that this case does not arise.

NOTE: In slave mode, it is important that the software programs the SL1-SL0 and Con2-Con0 bits to select a serial clock frequency greater than or equal to the transfer rate used on the MICROWIRE interface. A serial clock frequency of 1-5 MHz is suggested. Even though this serial clock will not drive the SCLK signal, it is used internally in the NS486SXL MICROWIRE interface to synchronize signals. If the internal serial clock is not programmed to meet this requirement, the MICROWIRE interface may not work as desired, in slave mode.

3.3.4.3 The Access.Bus Interface

Access.bus is a two wire interface that allows bidirectional communications between ICs. The two interface lines are the serial data line (SI), and the serial clock line (SCLK). Access.bus system configuration examples are shown in Figure 3-16 and Figure 3-17. As illustrated, Access.bus supports multiple masters and multiple slaves.

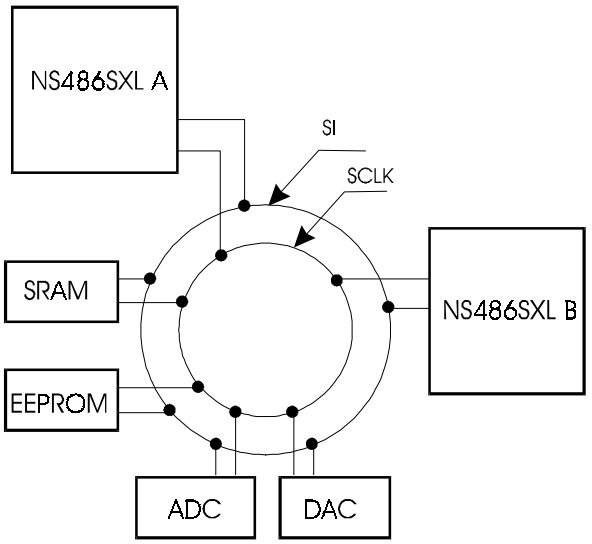


Figure 3-16 A Sample Access.bus System

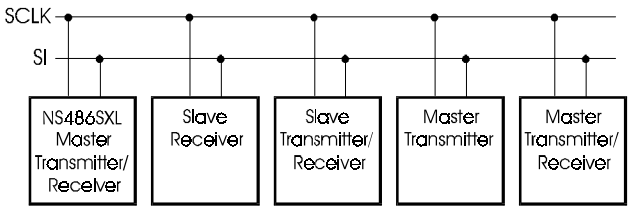


Figure 3-17 Multiple Resources

The Access.bus protocol includes software addressing and data transfer protocols in addition to hardware definitions.

The two serial lines (SI and SCLK) should be connected to a positive supply via a pull-up resistor and are to remain HIGH when the bus is not busy. Each device has a unique address and can operate as a transmitter and/or a receiver. (Note that some peripherals will only be receivers, and most controllers (including the **NS486SXL**) will be both transmitters and receivers.)

All Access.bus devices must drive the SI and SCLK lines with open-collector or open-drain drivers. In this manner, multiple devices may share these signals without any signal contention.

During data transfers, a device can be either a master or a slave. The initiating device is considered the master, it also generates the clock (SCLK) and the start condition for the transfer. The addressed device is considered the slave during the transfer. When the **NS486SXL** is initiating a data transfer with an attached Access.bus peripheral, it is the master and the transmitter. However, when the Access.bus peripheral in question responds and sends data to the **NS486SXL**, then the peripheral is the transmitter (even though it remains the slave) and the **NS486SXL** is the receiver (even though it remains the master). In other words, the master can be both a transmitter and a receiver. Likewise, a slave can be both a transmitter and a receiver. The key is that the initiator is the provider of the clock signal (SCLK) and is considered the present Access.bus master.

As shown in Figure 3-16 and Figure 3-17, it is possible to have more than one master on the bus. Because of this, an arbitration protocol has been included in the Access.bus implementation. The arbitration protocol depends on a wired-AND connection to the devices on the bus and clock synchronization. If two or more masters attempt to start a transfer at the same time, the master transmitting to the lowest address will win the arbitration.

3.3.4.4 Access.bus Data Transfers

Data is transferred during the high state of the serial clock (SCLK). Data can only change during the low state of SCLK. This is shown in Figure 3-18.

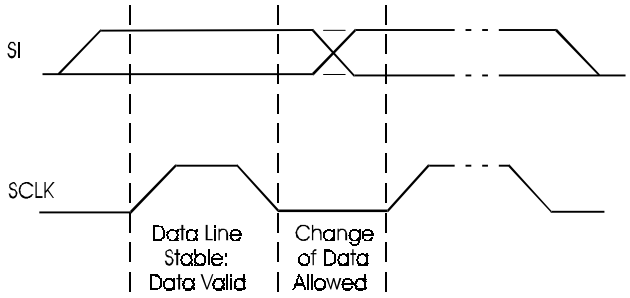


Figure 3-18 Bit Transfer

One data bit is transferred during each clock pulse. The data on the SI line must remain stable during the HIGH period to be valid. Transitions on SI during the entire high period of SCLK are interpreted as control signals. In this manner, a single data line is used to transfer both command/control information as well as data.

A high-to-low transition on SI while SCLK is high indicates a START condition as shown in Figure 3-19.

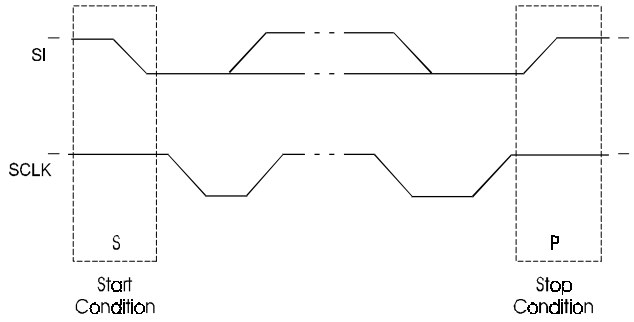


Figure 3-19 Start and Stop Conditions

Similarly, a low-to-high transition on SI while SCLK is high indicates a STOP condition.

The bus is considered to be busy after the START condition and free again after a certain time interval after the STOP condition. START and STOP conditions are generated by the present Access.bus master.

During a data transfer, the smallest unit of transfer is a byte. Any number of data bytes can be transferred in a given transfer sequence. Each byte is transferred with the most significant bit first and the least significant bit last. After the transfer of the least significant bit of each byte, an Acknowledge bit must be generated by the receiving device; as shown in Figure 3-20. During the Acknowledge bit time, the transmitting devices stops driving SI and the receiving device drives SI low. If the receiving device does not drive SI low during the Acknowledge bit time, SI will be pulled high by the pull-up resistor and the cycle will not be acknowledged.

Figure 3-20 Access.bus Data Transfer

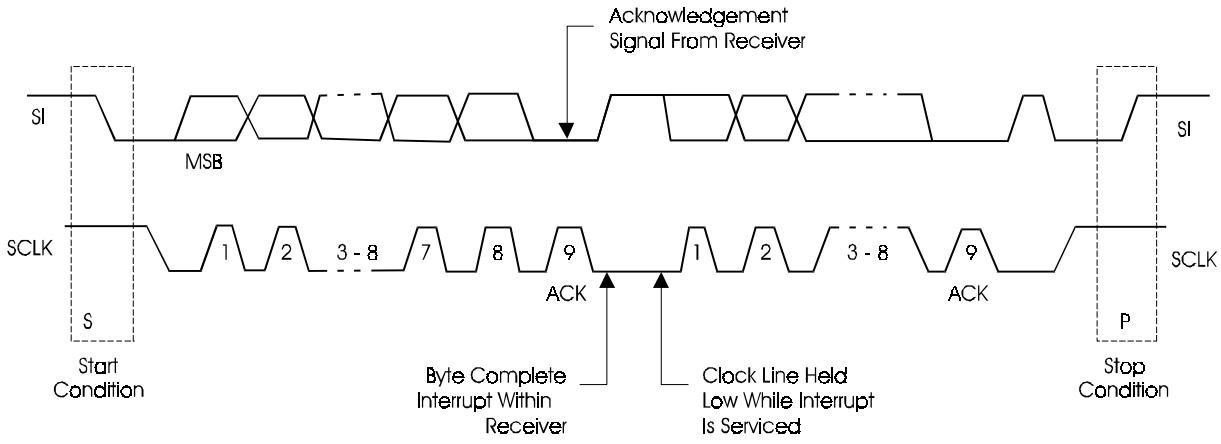
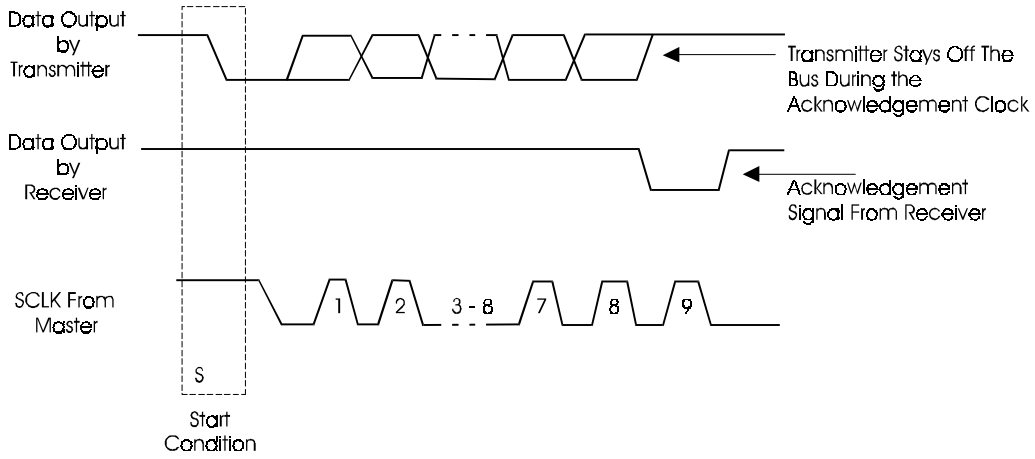


Figure 3-21 Access.bus Acknowledge Cycle



A slave receiver must generate an acknowledge after the reception of each byte, and a master must generate an acknowledge after the reception of each byte clocked out of the slave transmitter. If the receiving device cannot receive the data immediately, it can force the transmitter into a wait state by holding SCLK low. This can happen when the receiver is busy with some peripheral related task or is otherwise occupied with higher order system tasks. When this happens, the master, after a time-out of 2-3 msec., will abort the transfer and go to the idle state. This will also result in setting the Overflow bit in the MICROWIRE Control Register to a one. This type of cycle may also indicate that a device on Access.bus is in an unrecoverable condition and needs to be reset. User software should ensure that this condition is detected and resolved.

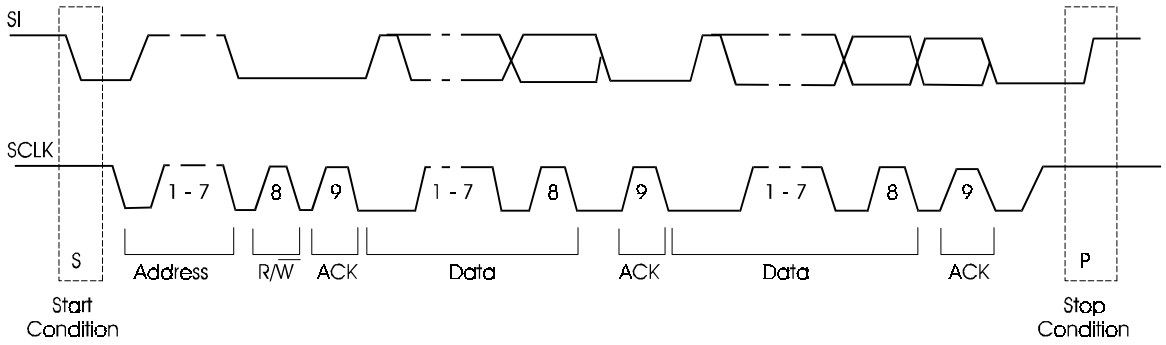
There are two exceptions to the “acknowledge after every byte” rule. The first is when the master is the receiver. It must signal an end of data to the transmitter by NOT generating an acknowledge to the last byte that is to be clocked out of the slave. This “negative acknowledge” still includes a SCLK pulse, (generated by the master), but SI will be pulled up by the pull-up resister. The second exception is when a slave sends a negative acknowledge to indicate that it can no longer accept additional data bytes. This may occur when the receiver FIFO is full, the receiver is otherwise occupied or the receiver is recovering from an error condition. If the slave receiver sends a negative acknowledge, the master will abort the transfer by generating the STOP condition.

3.3.4.5 Addressing Transfer Formats

Each device on the Access.bus must have a unique address. Before any data is transmitted, the master transmits the address of the slave being addressed. A slave device, once it recognizes its address, acknowledges the address. The **NS486SXL** Access.bus address is contained in the Own Address Register and may be modified via software. The **NS486SXL** Access.bus interface will also acknowledge an address of all zeros (called a "general call"), as required of all enabled Access.bus devices.

The address is the first seven bits after a START condition. The eighth bit is the direction (read = 1/ write=0, R/$\overline{W}$) indicator. A complete data transfer is shown in Figure 3-22 and shows the start condition, followed by a seven bit address, a one bit R/$\overline{W}$ indicator, a one bit Acknowledge, followed by the first data byte, a one bit Acknowledge and so on. A low-to-high transition on SI during a SCLK high period indicated the STOP condition and ends the transfer.

Figure 3-22 A Complete Access.bus Data Transfer

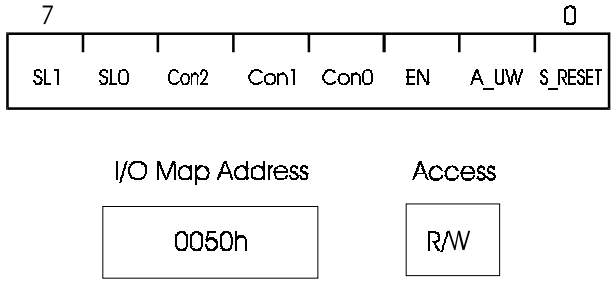


When the address is sent, each device connected to the Access.bus compares its address with the transmitted address. If there is a match, the device will consider itself addressed, and will generate an acknowledge. Depending upon the state of the R/$\overline{W}$ bit(read=1, write=0), the device will act as a transmitter (send data) or a receiver (receive data), respectively.

3.3.4.6 MICROWIRE/Access.bus Control Register (MACON)

The MICROWIRE interface supports a 0 - 1 MHz serial clock frequency and the Access.bus interface supports a 0-100KHz serial clock frequency. Bits 7-3 of this register may be programmed to generate these required serial clock frequencies.

Meanwhile, bits 2-0 allow the enabling of one of these interfaces, the selection of MICROWIRE or Access.bus and a soft reset bit, respectively.



Bits 7-6: SL1,2 — Selects the input OSCX1 clock divided by 8, 16, 32, or 64, as the prescaled serial clock.

Bits 5-3: Con2-0 — The prescaled serial clock generated by SL1 and SL2 will be further divided by (the value in these three bits plus one, all multiplied by 2). The result will be the serial clock to be used by either the MICROWIRE interface or the Access.bus interface (whichever is selected).

SL0 and SL1 are two select signals used to select a prescaled clock equal to the input OSCX1 clock divided by 8, 16, 32 or 64. This prescaled clock goes to a counter controlled by Con0, Con1 and Con2. With these five select bits, one can chose the OSCX1 clock divided by 16, 24, 32, 40, 48, 56, 64, 80, 96, 112, 128, 160, 192, 224, 256, 320, 384, 448, 512, 640, 768, 896 and 1024 as the clock source for either the MICROWIRE interface or the Access.bus interface (whichever is selected).

SL1	SL0	Input OSCX1 Clock divided by
0	0	8
0	1	16
1	0	32
1	1	64

Con0, Con1, Con2, SL1, and SL0 should be programmed according to the frequency of the input OSCX1 clock. These five bits determine frequency of the clock used in this block, and are used to optimize performance at differ oscillator speeds.

Con2	Con1	Con0	SL1 and SL0 selected prescaled Clock divided by
0	0	0	2
0	0	1	4
0	1	0	6
0	1	1	8
1	0	0	10
1	0	1	12
1	1	0	14
1	1	1	16

Bit 2: EN - When this bit is a one, it enables either the MICROWIRE interface or the Access.bus interface (whichever is selected by bit 1, A_UW). When this bit is a zero, neither the MICROWIRE interface nor the Access.bus interface will be enabled. When this bit is a zero, the SCLK, SI and SO pins may be used as modem control pins ($\overline{\text{DCD}}$, $\overline{\text{CTS}}$ and RI).

Bit 1: A_UW — When this bit is a zero the MICROWIRE interface is selected. When this bit is a one the Access.bus interface is selected.

Bit 0: S_RESET - Soft reset. When this bit is written to a one, it will reset the selected interface logic. Software must write a zero to this bit to clear it once it has been set to a one.

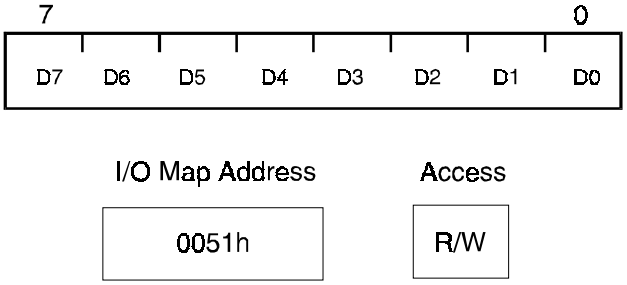
3.3.4.7 MICROWIRE Registers

MICROWIRE is a synchronous serial three-wire interface communication system that allows the **NS486SXL** to communicate with any other device that also supports the MICROWIRE interface.

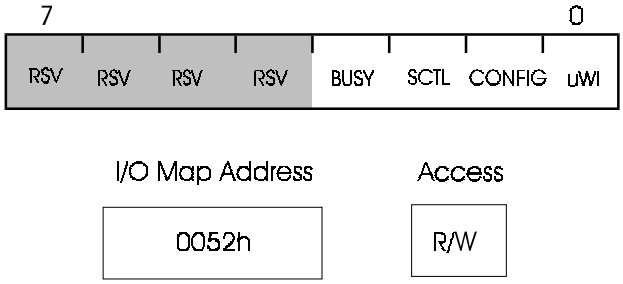
There are three 8-bit MICROWIRE registers:

3.3.4.7.1 Serial Input/Output Register (SIO)

An 8-bit shift register, called the SIO (Serial Input/ Output) register, is used for both transmitting and receiving data. In MICROWIRE systems, the most significant bit is transmitted(received) first and the least significant bit is transmitted(received) last. The **NS486SXL** CPU reads and writes to the SIO Register via an 8-bit parallel data bus.



3.3.4.7.2 MICROWIRE Control Register (uWCON)



Bits 7-4: Reserved.

Bit 3: BUSY — Software must set this bit to a one before the MICROWIRE interface can transmit or receive data. This bit will be cleared to a zero by hardware at the end of a MICROWIRE transfer cycle. While this bit is set to a one, it indicates that the MICROWIRE shift register is busy shifting data out/in.

Bit 2: SCTL — SCLK mode ConTroL. When this bit is a zero (standard SCLK mode), the output data on SO is clock out on the falling edge of SCLK and the input data on SI is sampled on the rising edge of SCLK.

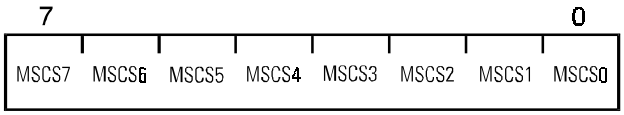
When this bit is a one (alternate SCLK mode), the output data on SO is clock out on the rising edge of SCLK and the input data on SI is sampled on the falling edge of SCLK.

Bit 1: CONFIG — Configuration bit. When one, the MICROWIRE interface will operate in Master mode. When zero, the MICROWIRE interface operates in slave mode.

Bit 0: uWI — MICROWIRE transmit/receive interrupt flag. This bit is set to a one at the end of SIO register shifting. This bit will be cleared to a zero by reading the SIO register or when the BUSY bit is set to a one. (This is a read only bit.)

3.3.4.7.3 MICROWIRE Slave Chip Select Register

The eight bits in this register select which pin will be used as the input MICROWIRE chip select, when the **NS486SXL** MICROWIRE interface is in slave mode. When the **NS486SXL** MICROWIRE interface is in master mode, the bits in this register will have no function. It is suggested that only one pin be selected as the input MICROWIRE chip select, but if more than one is selected, then an active low on any one of the selected pins will enable the **NS486SXL** MICROWIRE interface.



I/O Map Address	Access
EFC1h	R/W

Reset Value: 00000000h

Bits 7: MSCS7-MSCS0 — MICROWIRE Slave Chip Selection bits 7-0. When a one is written to a bit in this register it will select the corresponding pin listed below as the input MICROWIRE chip select. The input MICROWIRE chip select is an active-low signal which must be driven to a zero to select the **NS486SXL** MICROWIRE interface, when it is in slave mode.

Bit	Pin used as the input MICROWIRE chip select
7	$\overline{\text{DRV}}$
6	$\overline{\text{EACK}}$
5	IRQ[7]
4	Rx
3	$\overline{\text{CS}}$[4]
2	$\overline{\text{CS}}$[3]
1	$\overline{\text{CS}}$[2]
0	$\overline{\text{CS}}$[1]

Note: User software must also configure the chosen pin as a RIO input as well.

3.3.4.7.4 Access.bus Serial Interface

The two wires, serial data(SI) and serial clock (SCLK), carry information between the devices connected to the Access.bus. Each device is recognized by a unique address and can operate as either a transmitter or receiver, depending on the function of the device. It is a multi-master bus.

Four modes are supported:

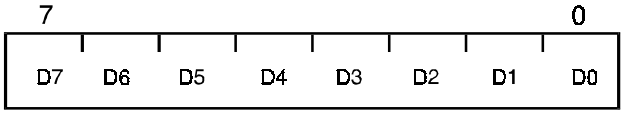
1) Master Transmit
2) Master Receive
3) Slave Transmit
4) Slave Receive

There are five registers inside the Access.bus module:

1) Serial Input/Output Data Register (SDA)
2) Access.bus Status Register
3) Access.bus Control Register 1
4) Access.bus Control Register 2
5) Own Address Register

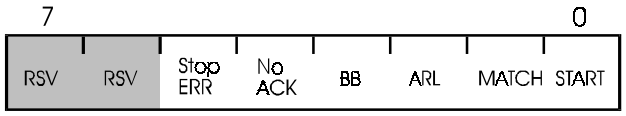
3.3.4.7.5 Serial Input/Output DATA Register (SDA)

An 8-bit shift register, called the Serial Input/Output register (SDA), is used for both transmitting and receiving data. The most significant bit is transmitted(received) first and the least significant bit is transmitted(received) last. The CPU reads and writes SDA via an 8-bit parallel data bus at I/O address 0053h.



I/O Map Address	Access
0053h	R/W

3.3.4.7.6 Access.bus Status Register



I/O Map Address	Access
0054h	Read

Reset Condition: 00h; all bits are set by hardware.

Bits 7,6: Reserved

Bit 5: Stop_ERR — Premature Stop Error Flag. This bit will be set to a one whenever a STOP condition is detected in the middle of a data transfer phase. When this error flag is set to a one, it indicates a possible system problem which must be addressed by the system software. It can be cleared by system reset, soft reset a START condition and by software writing a “1” to this bit.

Bit 4: No_ACK — No Acknowledge Error flag. This bit will be set to a one, in Master mode when a transmission is not acknowledged on the ninth clock. It can be cleared by system reset, soft reset a START condition and by software writing a “1” to this bit.

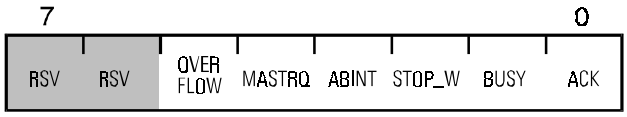
Bit 3: BB — Bus Busy flag. When set to a one, it indicates that the Access.bus is currently busy. It is set by a START condition and cleared by Reset or a STOP condition. (This is a read only bit.)

Bit 2: ARL — Arbitration Loss flag. When this bit is set to a one, it indicates that this device lost arbitration while trying to take control of Access.bus. It is cleared by system reset, soft reset, a STOP condition or software writing a “1” to this bit.

Bit 1: MATCH — Address match flag. In slave mode, this bit is set to one when the address byte (the first byte transferred) matches the 7-bit address in the Own Address Register. It is cleared by Reset, a START condition, STOP condition or software writing a “1” to this bit.

Bit 0: START flag. It will be set by hardware when a START condition is detected. It is cleared at the end of the first byte transfer, Reset or a STOP condition. It is cleared at the end of the first byte transfer, system reset, soft reset a STOP condition and by software writing a “1” to this bit.

3.3.4.7.7 Access.bus Control Register 1 (Write/Read)



I/O Map Address	Access
0055h	R/W

Reset Condition: 00h

Bits 7,6: Reserved.

Bit5: OVERFLOW — Overflow flag. This bit will be set to one when the serial clock (SCLK) is held low for more than 2-3 msec. In Master mode, when this bit is set to a one, the transfer should be aborted and the Access.bus interface logic should be reset by a soft reset (that is writing a one to the SR bit). This bit will be cleared by Reset or software writing a zero to this bit.

Bit 4: MASTRQ — Software must set this bit to a one to request control of the bus as a master.

Bit 3: ABINT — Access.bus interrupt flag. Will be set to a 1 at the end of a transfer by hardware. Must be cleared by software writing a zero to this bit.

Bit2: STOP_W — STOP request. In master mode, setting this bit to a one results in the generation of a STOP condition. This bit is cleared by Reset or by software writing a zero to this bit.

Bit 1: BUSY — Busy Transmitting flag. Software must set this bit to a one before the Access.bus interface can transmit or receive data. When this bit is a zero the SCLK signal is driven low, preventing any further Access.bus transfers.

Once set to a one, this bit will be cleared to a zero by hardware at the end

of an Access.bus transfer cycle from/to this logic. When this bit is set to a one, it indicates that Access.bus shift register is busy shifting data out/in.

Bit 0: ACK — Acknowledge flag. This bit is a status bit for transmit modes and a control bit for receive modes.

If this Access.bus interface logic is in Master Transmit mode and a transmitted byte is not acknowledged this bit will not be set; the Access.bus interface logic will generate a master abort (STOP condition). If this bit is set to a one, then software should clear it by writing it to a zero before the next transfer begins.

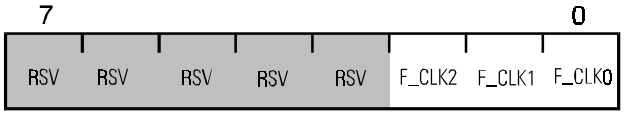
If this Access.bus interface logic is Slave Transmit mode and a transmitted byte is not acknowledged this bit will not be set; the Access.bus logic will simply stop its transfer and return to an idle mode. If this bit is set to a one, then software should clear it by writing it to a zero before the next transfer begins.

If this Access.bus interface logic is in either the Master or Slave Receive mode, this bit is under software control. Setting this bit to a one, will result in the generation of an Acknowledge at the appropriate time. Clearing this bit to a zero will result in no Acknowledge generation.

A Reset, STOP condition or software writing a zero to this bit, resets this bit to zero.

3.3.4.7.8 Access.bus Control Register 2 (Write/Read)

This register defines the number of clocks to use to filter out noise on SCLK and SI signals. An edge on either SCLK or SI will start a delay counter programmed by bits 2-0 of this register. If after the delay time, the value of SCLK or SI is the same (indicating that the edge was not a noise glitch), then at that time the change in SCLK or SI will be recognized.



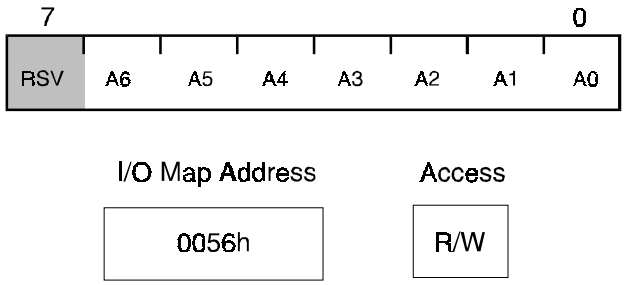
I/O Map Address	Access
0057h	R/W

Bits 7-3: Reserved.

Bit2-0: F_CLK — Filtering Clocks. These bits define the number of input OSCX1 clocks a new value on SCLK or SI must be maintained to be recognized. If the new value on SCLK or SI is less than this delay time, it will be filtered out and ignored.

Bits 2-0	OSCX1 Clock Periods
000	5
001	7
010 (reset value)	9
011	11
100	13
101	15
110	17
111	19

3.3.4.7.9 Own Address Register (Write/Read):



Bit 7: Reserved.

Bits 6-0: Loaded with the 7-bit Access.bus address. When addressed as a slave, the first seven bits transferred will be compared with this register to determine if this device is being addressed. This register should be programmed before the Access.bus interface is enabled, if the Access.bus interface is programmed to operate in Slave mode.

3.3.4.8 MICROWIRE/Access.bus Interface Programming Notes

If the current bus master wants to continue to use the Access.Bus to talk to another slave without possible arbitration, then Sr can be used. To program Repeat Start (Sr):

a) After the current transfer is finished (BUSY=0, ABINT=1); set MASTRQ=1 for repeat start
b) Set BUSY
c) Clear ABINT
d) Program SDA (Data Register) with the address of the slave.

During an Access.bus transfer, software must not write to the SDA register or the transferred data will be corrupted.

Do not reprogram the SCON (System Register) clock divide chain while the Access.bus is still busy.

The SIO register must be programmed only after the MICROWIRE interface is enabled.

STOP_W needs to be set before the last transfer finishes. A good time to set STOP_R is before the last byte transfer starts.

4.0 The System Bus

4.1 Bus Interface Unit (BIU)

The system bus, through a Bus Interface Unit (BIU) provides the interface for the CPU to on-chip peripherals, to memory and to external peripherals, such as a hard disk, a floppy disk or a modem. Furthermore, the BIU provides the necessary Chip Select logic and signals for those external peripherals. The BIU also provides five signals to support External Bus Masters. This allows an External Bus Master to have access to all external peripherals and memory, to the **NS486SXL**'s DRAM memory and to peripherals and system services that are internal to the **NS486SXL**. Wait states are supported through the Chip Select logic and by external peripheral feedback signaling. Though the bus does not fully emulate the ISA bus, it is capable of directly interfacing to most ISA peripheral control devices. See Section 4.2 for a more complete discussion of ISA-bus interfacing issuesThe BIU decodes all CPU accesses to internal and external resources and generates the appropriate control signals based on the type of access as well as the address of the access. The BIU always monitors the CPU local bus and if the access is to DRAM, the BIU will suspend operations and let the DRAM controller take over and generate the appropriate DRAM control signals. For all other accesses, the BIU will take control and generate the necessary control signals.

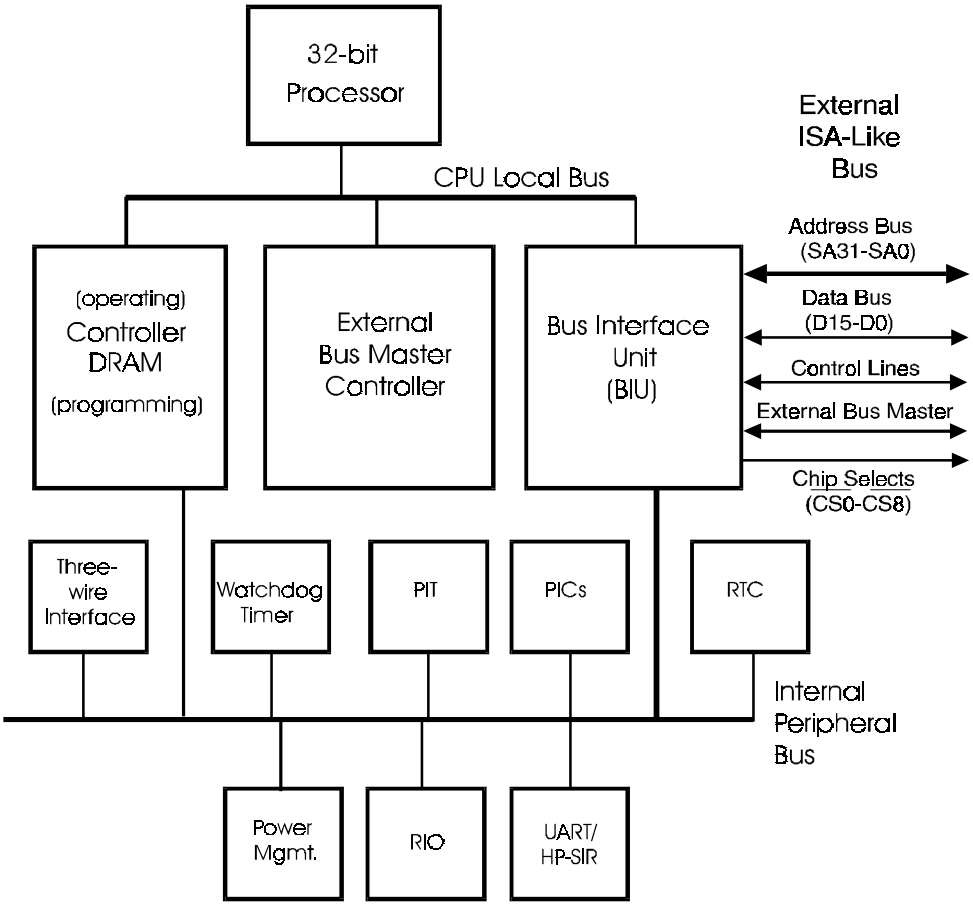


Figure 4-1 NS486SXL Internal Buses

The Bus Interface Unit contains a number of control registers (explained later in this section) that allow the user to:

1) Program chip selects (A set of Chip Select Address and Mask Registers, in conjunction with a Chip Select Enable Register and a Chip Select Access Type Register, control the generation of external chip select signals.)

2) Program which chip selects decode cacheable memory range(s) (via the Cacheable Chip Selects Register).

3) Program which chip selects are connected to 16-bit devices (via the 16-bit Logical Chip Select Register).

4) Extend the timing characteristics of control signals to the ISA-like external bus (via the Chip Select Access Time Registers).

5) Enable accesses to the following internal resources:

   a) The Three-wire interfaces (MICROWIRE and Access.bus)
   b) The UART
   c) The External Bus Master Controller
   d) The Interrupt Controllers
   e) The Programmable Interval Timer
   f) The Real Time Clock

(via the Bus Interface Unit Control Registers 1 and 2).

6) Monitor internal bus cycles to assist in tracing and debugging internal accesses (via a bit in the Bus Interface Unit Control Register 1).

4.1.1 Internal Peripheral Accesses

The BIU decodes all CPU accesses to internal peripherals and generates the appropriate internal address and control signals. In the default configuration of the BIU, no external indication of these internal bus cycles will be made; this mode of operation provides for lower power consumption. However, if the Internal Cycle Debug bit (ICD, bit 0 in the BIU Control Register 1) is set to a one, an external bus cycle will be performed at the same time the internal access is being made; this mode of operation provides information indicating when accesses are being made to the various internal peripherals. Furthermore, during reads from internal peripherals, with ICD = 1, the value being read by the CPU will be driven out on the SD31-0 data bus pins. It is important that no external peripheral drives SD31-0 during these bus cycles.

Accesses are made to internal peripherals in three CPU T-states; this corresponds to the shortest possible external bus access timing.

4.1.1.1 NS486SXL Peripheral Control Registers

The peripheral functions within the **NS486SXL** may be broken into two general categories: **NS486SXL** specific functions and PC industry standard functions.

The control registers for all of the **NS486SXL** specific functions reside in the I/O address range EF00h-EFFFh. These I/O addresses will always be accessible by the **NS486SXL** CPU and must never be used by externally connected peripherals.

The PC industry-standard functions have historically resided in certain I/O address ranges. To remain consistent with the historical implementation, the **NS486SXL** has whenever possible mapped the PC industry standard functions into their historical I/O address ranges.

An issue with using the historical I/O address ranges is that if one of these peripherals is not to be used, its I/O address range may conflict with an I/O address range of an external peripheral. To eliminate this **potential** problem, following an IC reset, the access to all PC industry standard peripherals will be disabled. A one must be written to the appropriate enable bits in the BIU Control Register 1 and the BIU Control Register 2, to enable access to each internal peripheral. When an internal peripheral is disabled, accesses to its address range will become external bus cycles.

The BIU Control Registers 1 and 2, only enable/disable the ability of the CPU to access these internal peripherals and does not enable/disable the function itself. This fact is especially important with regards to the Real Time Clock (RTC), since it may be battery backed and will continue to operate even when sys-

tem power is lost. When the RTC is battery backed, the RTC will continue to operate after an IC reset, but the CPU must enable its access to the internal RTC via the BIU Control Registers 1 and 2 before the CPU will be able to read and write to the RTC.

4.1.1.2 Global Enable

Bit 2 (GPE) of BIU Control Register 2 (I/O address EF01h) is a global access enable bit for PC industry-standard internal peripherals. When GPE is a zero, the CPU will not be able to read or write these internal peripherals. When GPE is a one, the access enable bit associated with each peripheral will determine if the peripheral may be accessed or not.

4.1.1.3 Accessing The Internal UART

The **NS486SXL** internal UART can be configured to reside in the four different industry standard COM Port address ranges. Bits 7 and 6 (UART_S1 and UART_S0, respectively) of the BIU Control Register 2 determine which of the four possible I/O address ranges is used.

UART_S1	UART_S0	I/O Address Range
0	0	03F8h-03FFh (COM1)
0	1	02F8h-02FFh (COM2)
1	0	03E8h-03EFh (COM3)
1	1	02E8h-02EFh (COM4)

4.1.1.4 Accessing Other Internal Peripherals

The PC industry standard peripherals which only map into a single set of address ranges include the Three-Wire Interfaces, the Timer, the Real Time Clock, the Interrupt Controllers. When they are enabled, these peripherals map into the following I/O address ranges.

	I/O Address Range
Three-Wire	0050h-0057h
Timer	0040h-0047h
Real Time Clock	0070h-0071h
Interrupt Controllers	0020h-0021h
“	00A0h-00A1h

Controller is not to be used), then the BIU will not insert the two command delays; instead the BIU will perform the appropriate bus cycle without delay.

4.1.2 Interrupt Acknowledge Cycles

The BIU decodes all interrupt acknowledge cycles for both internal and external interrupt controllers. The Interrupt Acknowledge signal ($\overline{\text{INTA}}$) strobes active low during each interrupt acknowledge cycle. Simultaneously, the three most significant bits of the System Address (SA31-29) will be driven with the three Cascade Line signals (CAS2-0, respectively) from the internal master interrupt controller. In this manner an external interrupt controller can determine if it is suppose to respond with the appropriate interrupt vector.

When a cascaded external interrupt controller provides the appropriate interrupt vector (during the second $\overline{\text{INTA}}$ strobe) in an interrupt acknowledge sequence, the BIU reads the vector from the SD7-0 pins and presents it to the CPU as required.

Likewise, when an internal interrupt controller provides the appropriate interrupt vector (during the second $\overline{\text{INTA}}$ strobe) in an interrupt acknowledge sequence, the BIU will read that 8-bit vector and present it to the CPU as required.

Refer to Appendix A: External Interrupt Controller for information regarding how a cascaded external interrupt controller should be connected to **NS486SXL**.

4.1.3 HALT Cycles

The BIU decodes and acknowledges a HALT cycle from the CPU, but no other action will be taken.

4.1.4 SHUTDOWN Cycles

The BIU decodes and acknowledges a SHUTDOWN cycle from the CPU and when the SHutdown Reset bit (SHR, bit 1 of the BIU Control Register 1) is a one, an internal CPU reset will be performed. When SHR is a zero, the BIU will decode and acknowledge a SHUTDOWN cycle, but no CPU reset will be performed.

When a SHUTDOWN cycle results in the generation of a CPU reset, only the CPU will be reset, the on chip peripherals will not be reset and the RESET and $\overline{\text{RESET}}$ pins will remain in their inactive states.

4.2 External Bus Cycles

DRAM accesses are more frequent than any other, so to optimize the DRAM access times, the logic is designed to anticipate a DRAM access cycle. When a DRAM access is made, the minimum number of T-states are used. However, when a non-DRAM access is performed the first T-state of each cycle allows control of the access to be transferred from the DRAM Controller to the Bus Interface Unit. Figure 4-6 shows the timing diagram associated with the shortest possible non-DRAM read cycle.

4.2.1 ISA-like Bus Operation

NS486SXL's External Interface Bus can be described as ISA-like because **NS486SXL's** external bus can operate with most ISA bus peripherals and it meets many of the most widely used specifications of the ISA bus. It is not exact, however, and individual usage must be examined to ensure adequate compatibility.

Figure 4-2 ISA-like Bus Cycle Timing shows the basic functional signal relationships of the **NS486SXL** ISA-like bus.

To insert extra wait states into a ISA-like bus cycle, an external device may pull the ready pin (RDY) inactive low. The BIU will insert wait states indefinitely, until the RDY signal returns high. Typically, in the system design, the RDY signal is pulled up with a resistor and multiple devices can drive this signal with open-collector, open-drain or TRISTATE drives; thus allowing multiple devices the ability to insert additional wait states via the same input signal pin. Figures 4.11-4.13 illustrate the insertion of wait states using the RDY feedback signal.

The **NS486SXL's** Bus Interface Unit (BIU) also accepts a feedback signal $\overline{CS16}$, which functions similarly to the $\overline{IOCS16}$ and $\overline{MEMCS16}$ signals on the ISA bus. One important difference between the **NS486SXL's** $\overline{CS16}$ and the ISA bus signals is that the **NS486SXL** samples $\overline{CS16}$ near the end of the command strobe (i.e. $\overline{MEMR}$, $\overline{MEMW}$, $\overline{IOR}$, $\overline{IOW}$), rather than the leading edge as in the ISA bus. This is done to allow the user to gate $\overline{CS16}$ with the command strobe for qualification. When, the $\overline{CS16}$ signal is driven active low, the BIU will perform a 16-bit device access for the present cycle. On the other hand, if $\overline{CS16}$ is inactive high (indicating an 8-bit device), the BIU will steer the ISA-like bus to perform 8-bit transfers on the lower byte of the System Data Bus (SD7-0). The BIU will also perform any required 16-bit to two 8-bit cycle translations, similar to a standard ISA bus controller.

Note: The Programmable 16-bit Logical Chip Select Register (I/O address EF57h) contains control bits that affect 16-bit accesses. When any Chip Select bit in this register is set to 1, its associated logical chip select access will be treated as a 16-bit access regardless of the state of the $\overline{CS16}$ signal. This is to provide support for memory devices (EPROM's, SRAMs, etc.) that would require extra external logic to generate the appropriate feedback signal on the $\overline{CS16}$ pin. This allows 16-bit memories to be connected to the **NS486SXL** with no additional external logic.

Figure 4-14 depicts the timing of a 16-bit access being broken into two 8-bit cycles (the minimum cycle times are assumed for this figure).

Other ISA-like Bus Features

Any of the standard ISA Bus Interrupt Requests (i.e. IRQ15, IRQ14, IRQ12, IRQ11, IRQ10, IRQ9, IRQ7, IRQ6, IRQ5, IRQ4 and IRQ3) may be steered to the six interrupt pins provided on the **NS486SXL** (refer to the Interrupt Source Selection section). The **NS486SXL** only supports six interrupt requests at any one time, so only six of the eleven of the ISA bus interrupt requests can be supported at the same time.

The 'x86 architecture supports byte, word, and double word accesses to memory. The DRAM controller automatically handles these accesses to DRAM, and the BIU will convert 32-bit accesses into sequential 8 or 16 bit accesses. However, external 16-bit memory must support both 8 and 16 bit accesses. The SA0 and $\overline{SBHE}$ signals can be used during memory and I/O cycles to determine the width of an access. The 16-bit memory system should interpret these signals as follows:

SA0	$\overline{SBHE}$	Meaning
0	0	16-bit transfer
1	0	8-bit transfer (odd/ high byte)
0	1	8-bit (even/low byte)

4.2.2 ISA-like Bus Timing

This section provides information about the ISA-like bus timing.

> **Note:** The numbers used in this text are approximations, and are provided only to give the user a better understanding of the design.

Important ISA and **NS486SXL** terms include:

Command Delay Time — This is the amount of time from a valid address until a command strobe (i.e. $\overline{IOR}$, $\overline{IOW}$, $\overline{MEMR}$ or $\overline{MEMW}$) goes active low. See Section 4.4.6 on page 74 and Section 4.5.7 on page 83.

Access Time — This is the amount of time that a command strobe (i.e. $\overline{IOR}$, $\overline{IOW}$, $\overline{MEMR}$ or $\overline{MEMW}$) is active low. See Section 4.4.6 on page 74 and Section 4.5.7 on page 83.

Address Hold Time — This is the amount of time that a valid address is held after the trailing edge of a command strobe (i.e. $\overline{IOR}$, $\overline{IOW}$, $\overline{MEMR}$ or $\overline{MEMW}$).

Data Hold Time — This is the amount of time that the write data is held after the trailing edge of a write command strobe (i.e. $\overline{IOW}$ or $\overline{MEMW}$).

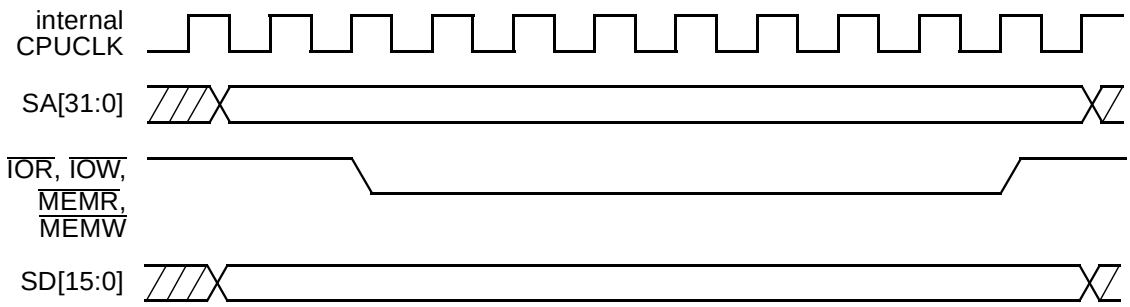
RDY Extension Time — This is the amount of time from the RDY signal going active high until the trailing edge of a command strobe (i.e. $\overline{IOR}$, $\overline{IOW}$, $\overline{MEMR}$ or $\overline{MEMW}$). See “Boot ROM Access Time Register” on page 85.

To be the most ISA-Like, the following options should be selected:

1) Maximum number of Command Delay States (one)
2) Maximum number of Programmed Wait States (seven)
3) Maximum number of RDY Extension Clock Periods (two)

The timing diagram in Figure 4-2 shows the result when RDY remains active high.

Figure 4-2 ISA-like Bus Cycle Timing



In general the **NS486SXL's** ISA-like External Interface bus may be programmed to support a wide range of ISA Bus components and boards.

The following numerical calculations assume an internal CPU operating frequency of 25 MHz.

NS486SXL Command Delay Time:

= (Programmed # of Command Delays + 0.5) X (CPUCLK period)
= ~1.5 CPUCLK periods
= 1.5 (40 nsec)
= 60 nsec

For ISA Command Delay time:

= ~40 nsec for 16-bit boards or devices
= ~100 nsec for 8-bit boards of devices

NS486SXL Access Time:

= (Programmed # of Wait States + 1) X (CPUCLK period)
= ~8 CPUCLK periods
= 8(40 nsec)
= 320 nsec

For ISA Access Time:

= ~170 nsec for 16-bit IO boards or devices
= ~225 nsec for 16-bit Memory boards or devices
= ~500 nsec for 8-bit boards or devices

NS486SXL Address Hold Time:

= ~1 CPUCLK period (always)
= 40 nsec

For ISA Address Hold Time:

= ~40 nsec

NS486SXL Data Hold Time:

= ~1 CPUCLK period (always)
= 40 nsec

For ISA Data Hold Time:

= ~35 nsec

NS486SXL RDY Extension Time:

= (Programmed # of RDY Extension Clock Periods) X(CPUCLK period)
= ~2 CPUCLK periods
= 2(40 nsec)
= 80 nsec

For ISA RDY Extension TIme:

= ~120 nsec

4.3 Device Configuration at RESET

At the end of a POWERGOOD (PWGOOD) reset, the **NS486SXL** must select each of the following functions to be supported:

1) 8-bit Boot-up ROM BIOS
   vs. 16-bit Boot-up ROM BIOS

2) The **NS486SXL** operates in normal mode
   vs. the **NS486SXL** will TRI-STATE all of its output drivers because it recognizes that an In Circuit Emulator (ICE) pod is attached to the system.

These two functions must be determined before any code is fetched or executed. To achieve this requirement, two pins which are normally outputs or reserved ($\overline{\text{SBHE}}$, and SA[0]), become inputs while the PWGOOD input signal is inactive low. Upon the rising edge of the PWGOOD signal, the values on these pins will be latched into two separate registers, which control the above functions as discussed below. These two functions are considered "strapping options" because they are set immediately upon coming out of reset.

4.3.1 Boot-up ROM BIOS Size

When the PWGOOD signal is inactive low (indicating the **NS486SXL** is being reset), the $\overline{\text{SBHE}}$ pin becomes an input. If $\overline{\text{SBHE}}$ is pulled high with a 10 Kohm resistor, an internal latch will latch in a one upon the rising edge of PWGOOD, that selects 16-bit Boot-up ROM BIOS support. On the other hand, if $\overline{\text{SBHE}}$ is pulled low by a 10 KOhm resistor, the internal latch will latch in a zero, selecting 8-bit Boot-up ROM BIOS support.

When 16-bit Boot-up ROM BIOS is selected, all memory accesses to the boot-up address segment FFFF0000h-FFFFFFFFh will be performed as 16-bit accesses with all even address code bytes fetched on SD[7:0], while all odd address code bytes would be fetched off of SD[15:8]. This mode of operation fetches code twice as fast as 8-bit Boot ROM BIOS, but it also requires either two 8-bit EPROMs or a 16-bit wide EPROM.

When 8-bit Boot-up ROM BIOS is selected, all memory accesses to the boot-up address segment FFFF0000h-FFFFFFFFh will be performed as 8-bit accesses with all code bytes fetched on SD[7:0]. This reduces the component count. Also it should be noted that in some systems, once the **NS486SXL** has booted up, it will no longer access this memory range, so doubling the boot-up time may not have any impact on the system performance at other times.

The 10 KOhm pull-up or pull-down resistor will have no impact on the normal operation of the **NS486SXL** after PWGOOD becomes active high. Following a small delay after PWGOOD rises, the $\overline{\text{SBHE}}$ signal will become a driven output in the **NS486SXL** normal mode of operation.

4.3.2 Normal Mode vs. ICE Mode

When the PWGOOD signal is inactive low (indicating the **NS486SXL** is being reset), the SA[0] pin becomes an input. If it is left floating, an internal latch will latch in a one upon the rising edge of PWGOOD, that selects **NS486SXL** normal mode (there is a weak pull-up resistor internal to the NS486SXL). On the other hand, if SA[0] is driven low during this time the internal latch will latch in a zero. If a zero is latched from SA[0] and the $\overline{\text{TEST}}$ signal is asserted (low) then the **NS486SXL** will enter ICE TRI-STATE Mode.

NS486SXL Normal Mode is the normal operation of the **NS486SXL** that is discussed throughout this Data Sheet.

NS486SXL ICE TRI-STATE Mode results in all of the output drivers on the **NS486SXL** being placed in TRI-STATE. An In-Circuit-Emulator should be the only thing that drives SA[0] low via its pod connection. The $\overline{\text{TEST}}$ pin should be pulled high through a resistor in the target design, so that the ICE can assert it low as well. An ICE may now be directly connected to a board under development without removing the **NS486SXL** in that system.

After a small delay from when PWGOOD rises, the SA[0] signal will become a driven output in the **NS486SXL** normal mode of operation.

4.4 Chip Select Logic

There are nine external chip select pins on the **NS486SXL** ($\overline{CS[0]}$ - $\overline{CS[8]}$). Chip Select 0 ($\overline{CS[0]}$) should be connected to the system Boot ROM. The other eight programmable chip selects may be used to interface to external slave devices connected to the **NS486SXL** interface bus. Chip Select 0 ($\overline{CS[0]}$) is a unique chip select that is designed to generate the chip select for the system boot ROM. As a result, $\overline{CS[0]}$ will always go active for any CPU access to the CPU's reset address segment (that is the most significant 64 Kbytes of CPU memory address, FFFF0000h - FFFFFFFFh). Additional memory ranges may be added to $\overline{CS[0]}$ via the programmable chip select logic. Reference the following sections and the Boot ROM Selection Register for more information regarding how the additional memory range(s) may be programmed to the $\overline{CS[0]}$ Chip Select pin.4.3 Chip Select Logic

4.4.1 Boot ROM Data Bus Size

The $\overline{SBHE}$ pin of the **NS486SXL** is normally an output, but during a system reset (when PWGOOD equals zero), $\overline{SBHE}$ will be an input to the **NS486SXL**. Externally, this signal should be pulled up or down via a 10 Kohm resister. When the PWGOOD signal rises, the value on the $\overline{SBHE}$ pin will be latched into an internal strapping latch. The value of this $\overline{SBHE}$ strapping latch will determine if the **NS486SXL** will access the Boot ROM memory address range (FFFF0000h-FFFFFFFFh) with 16-bit cycles or 8-bit cycles.

$\overline{SBHE}$ Strapping Latch	Boot ROM Size
0	8-bit
1	16-bit

An 8-bit ROM size indicates that each byte of the Boot ROM (memory address range FFFF0000h-FFFFFFFFh) will be accessed via the low byte of the system data bus (SD7-0).

A 16-bit ROM size indicates that all even address bytes of the Boot ROM will be accessed via the low byte of the system data bus (SD7-0) and all of the odd address bytes of the Boot ROM will be accessed via the high byte of the system data bus (SD15-8). This will allow the **NS486SXL** to access twice as many bytes as it would in the 8-bit access mode.

By implementing this feature, the **NS486SXL** can boot-up out of 8-bit or 16-bit Boot ROM without any additional hardware to generate the external $\overline{CS16}$ feedback signal.

4.4.2 Other Chip Selects' Data Bus Size

The logic associated with the programmable chip selects also provides an 8-bit register which allows software to select whether each logical chip select supports 16-bit data size independent of the $\overline{CS16}$ feedback signal. This 8-bit register is located at I/O address EF57h and is called the Programmable 16-bit Logical Chip Select Register. For more information regarding the function of each bit of this register, refer to its register description in section 4.5.10 on page 89.

By providing this feature, the **NS486SXL** provides a way that the system designer can connect 16-bit external memories (EPROMs or SRAMs) to the ISA-like bus without any additional hardware to generate the external $\overline{CS16}$ feedback signal.

The system designer may connect a 16-bit memory to the ISA-like bus and have the software program a logical chip select to decode the appropriate address range for the memory, as well as setting the appropriate bit in the Programmable 16-bit Logical Chip Select Register. As a result the **NS486SXL** will always generate the appropriate memory access signals to the 16-bit memory; placing all of the even byte data on SD7-0 and all of the odd byte data on SD15-8.

4.4.3 Programmable Chip Select Architecture

NS486SXL chip select pins ($\overline{CS[8:0]}$) are driven using a two step process. First, a "Logical" chip select is created using an address range and the access type (I/O or Memory). Then a single logical chip select or combination of logical chip selects can be used to drive a given external Chip Select pin. This allows a single Chip Select pin to select a device such as the National PC87332 SuperI/O™ multifunction peripheral controller. These devices contain multiple I/O controllers (dual UARTs, a floppy disk controller, an ECP port, and a hard disk controller) in a single package. With the **NS486SXL** chip select structure, it can

activate a single Chip Select pin with accesses to several different I/O address ranges. This process can be seen in Figure 4-3 and Figure 4-4.

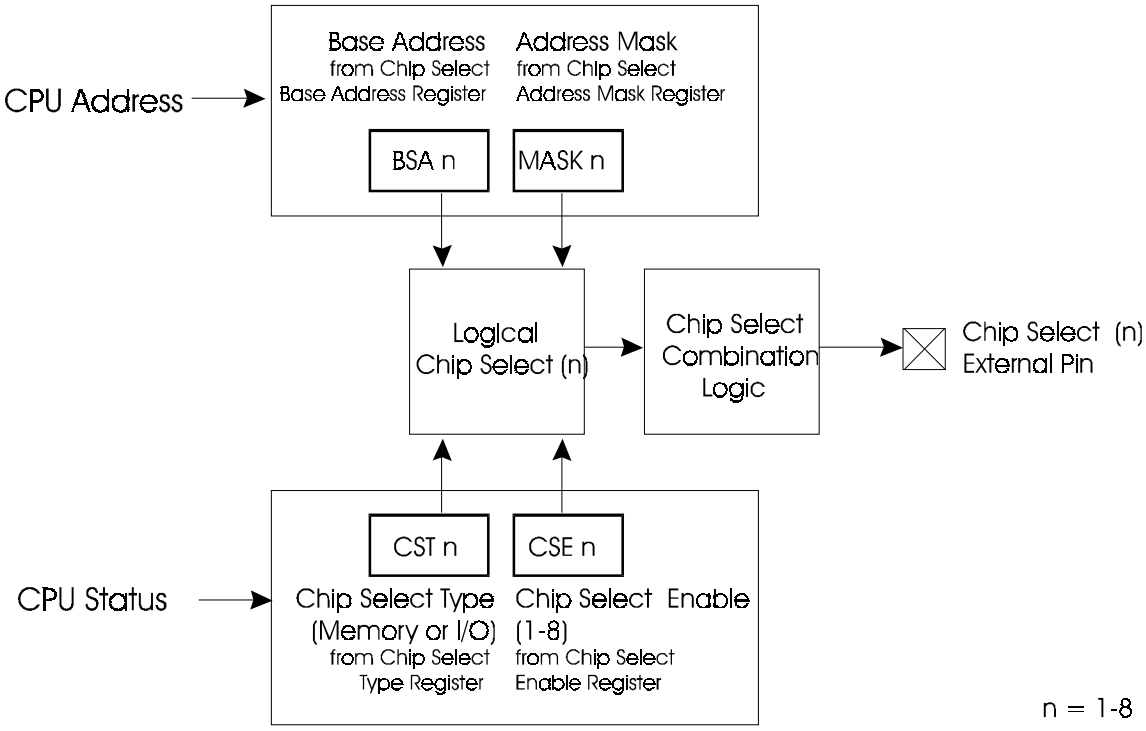


Figure 4-3 Logical Chip Selection

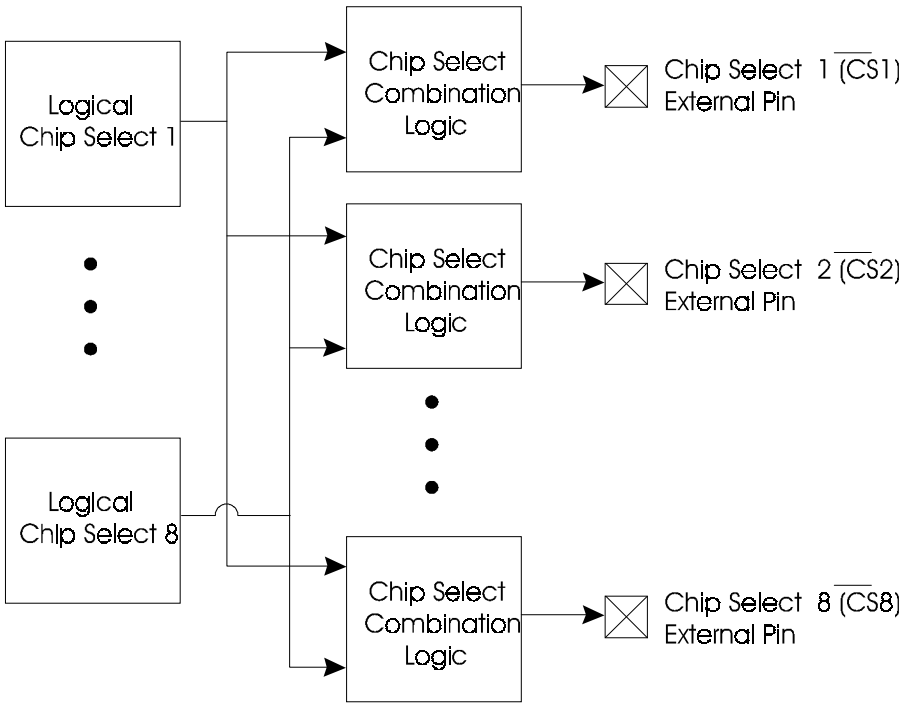


Figure 4-4 External Chip Selection

4.4.4 Programming Logical Chip Selects

When programming a Logical Chip Select, two functions must be programmed:

1) The type of access (I/O or Memory) associated with the chip select.
2) The address range associated with the chip select.

The user must program the 8-bit Chip Select Type Register (I/O address EF03h) to set the type of access associated with each Logical Chip Select. Bit 0 corresponds to Logical Chip Select 1, bit 1 corresponds to Logical Chip Select 2, and so on, up to bit 7 and Logical Chip Select 8. If a bit of the Chip Select Type Register is a 0, then the Logical Chip Select associated with that bit will decode I/O addresses; if that bit is a 1, then the associated Logical Chip Select will decode Memory addresses.

To program the active address range associated with a Logical Chip Select, the user must program its 32-bit Base Address Register (at I/O addresses EF04h-EF23h) and its 26-bit Address Mask Register (I/O addresses EF24h-EF43h).

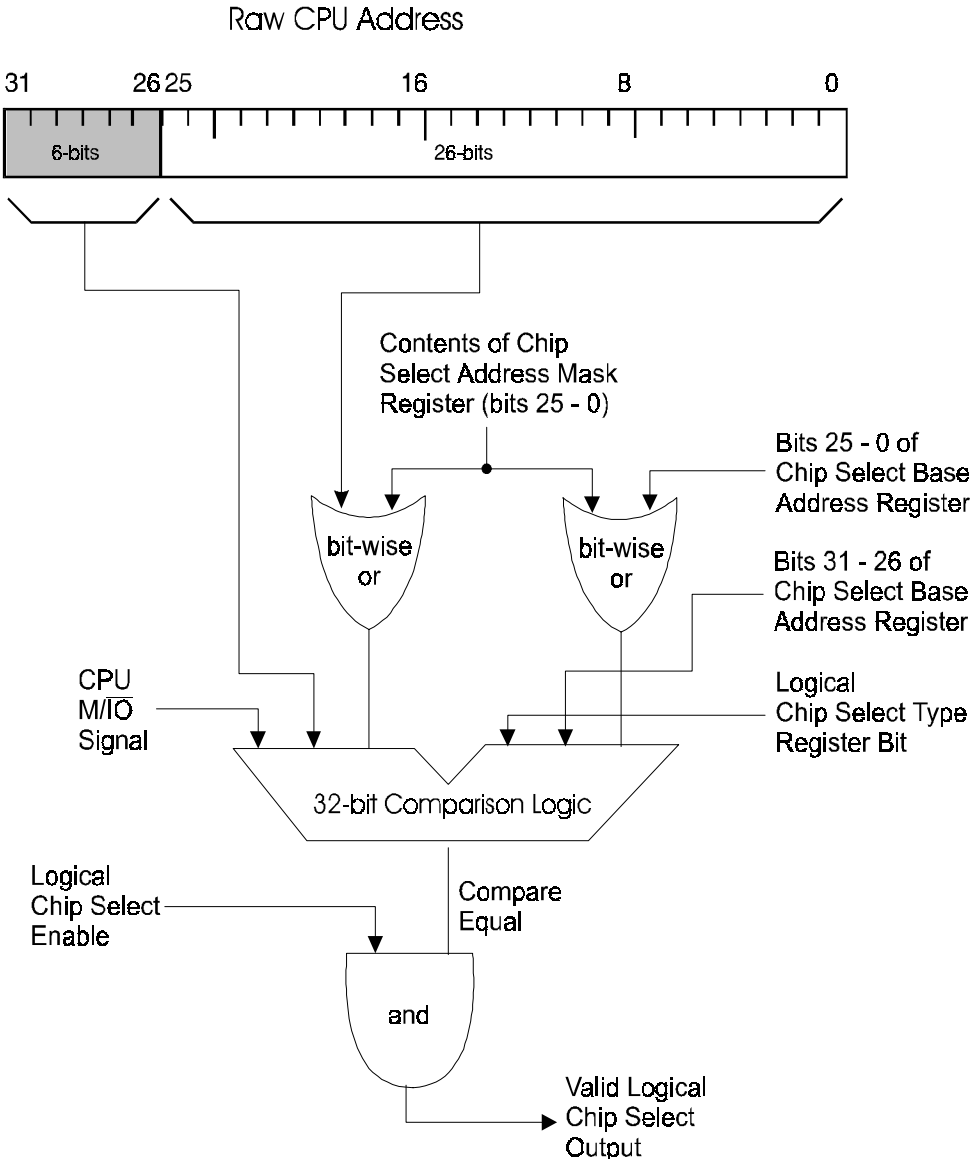


Figure 4-5 Address Masking

The Base Address Registers provide a complete 32-bit address that is compared to CPU generated addresses. The Address Mask Registers are 26-bit in size (since only the lower 26 address bits are driven off-chip) and they define the size or range of addresses associated with each Logical Chip Select. A zero in a bit position in the Address Mask Register indicates that the corresponding address line must match the CPU address. A one in a bit position in the Address Mask Register indicates that address bit is a "*don't care*". This means that these address bits that toggle throughout a given address range are "masked off" and a valid chip select is generated whenever any of these addresses are selected.

This is shown in Figure 4-5 and explained in the following examples.

EXAMPLE 1:

Logical Chip Select Active Addresses:
0070h -0071h
Base Address Register:
0000_0070h
Address Mask Register:
0000_0001h

or

Base Address Register:
0000_0071h
Address Mask Register:
0000_0001h

The user should note from the above example that the least significant address bit is a "*don't care*"; this is indicated by the 1 in the least significant bit of the Address Mask Register. As a result, it does not matter whether the Base Address Register is loaded with either 0000_0070h or 0000_0071h (the least significant bit being a one or a zero).

It is suggested (but not required) that any bit programmed as a "*don't care"* in the Address Mask Register have its corresponding Base Address bit programmed as a 0.

EXAMPLE 2:

Logical Chip Select Active Addresses
F_0000h - F_7FFFh
Base Address Register:
000F_0000h
Address Mask Register:
0000_7FFFh

The 15 least significant bits of the Address Mask Register are all 1's, so the 15 least significant bits of the address decode are "*don't cares"*. So, for simplicity and clarity, the 15 least significant bits of the Base Address Register are all 0's.

Contrast the following example to the previous example.

EXAMPLE 3:

Logical Chip Select Active Addresses:
F_8000h - F_FFFFh
Base Address Register:
000F_8000h
Address Mask Register:
000_7FFFh

The only modification is that bit 15 of the Base Address Register was changed from a 0 to a 1. As a result the decode of the active address range moved to the next 32 Kbyte range.

The user must remember that the bits programmed as 1's in the Address Mask Register, indicate only which address bits are "*don't cares"*. They do **not** indicate the number of addresses to be decode. The following example may help in this regard:

EXAMPLE 4:

Logical Chip Select Active Addresses:
0060h and 0064h
Base Address Register:
0000_0060h
Address Mask Register:
000_0004h

Only address bit 2 is a "*don't care"*; indicated by a 1 in bit 2 of the Address Mask Register (04h). However, address bits 1-0 are not "*don't cares"*; as a result the chip select will only be active for addresses 0060h and 0064h, and will not be active for addresses 0061h - 0063h.

Furthermore, to decode a single address location, one should program the Address Mask Register with all 0's.

EXAMPLE 5:

Logical Chip Select Active Address
1234_5678h
Base Address Register:
1234_5678h
Address Mask Register:
000_0000h

None of the address bits are *"don't cares"*, so the address generated by the CPU and the address in the Base Address Register, must exactly match for the Logical Chip Select to become active.

4.4.5 Combining Chip Selects

One limitation associated with each logical chip select is the fact that only a single I/O or memory address range may be selected by each logical chip select.

To overcome this limitation, the **NS486SXL** provides circuitry which allows the user to combine multiple logical chip selects into a single signal, which is then driven out onto one of the generic chip select pin ($\overline{CS[1]}$ - $\overline{CS[8]}$).

An 8-bit register (External Chip Selection Registers 1 through 8, at I/O addresses EF45h - EF4Ch) associated with each generic chip select pin ($\overline{CS[1]}$ - $\overline{CS[8]}$) defines which logical chip select(s) are to be combined to generate the signal driven onto the output chip select pin.

This provides the ability to generate a chip select with a minimum of a single memory or I/O address range; or, on the other hand it provides the ability to generate a chip select with a maximum of eight different memory and/or I/O address ranges.

Eight logical chip selects provide a maximum of eight address ranges, that can be combined in any desired fashion to drive the generic chip select pins ($\overline{CS[1]}$ - $\overline{CS[8]}$). Refer to Figure 4-4.

It should be noted that if the user programs a chip select decode to coincide with either DRAM memory or an enabled internal peripheral, the external command strobes will not go active when such an access takes place, but the associated chip select signal will still go active.

4.4.6 Programmable Access Timing

The Chip Select Access Time Registers 1 through 4 (at I/O address locations EF50h - EF53h) provide the means to program access timing parameters. Each Logical Chip Select has three bits which determine the number of additional programmed wait states associated with an access to its decoded locations. The number of added wait states may be any number between 0 and 7; resulting in a total of between 1 and 8 access states per cycle. More wait states may be added by any external device driving the Ready signal (RDY) inactive low. RDY is sampled at the end of the last programmed wait state, and if it is inactive low, another wait state will be added to the cycle. The Bus Interface Unit will continue to insert wait states until it samples RDY active high, then the access will be terminated.

Also in the Chip Select Access Time registers there is a bit associated with each Logical Chip Select which determines if the default command delay value is zero or one clock period. At higher frequencies one command delay may be required, otherwise the chip select may have negative setup time.

An External Bus cycle with the maximum number of wait states (seven) and the maximum number of inactive command states (one) will be performed if the access is not to DRAM, an internal peripheral or associated with a chip select decode.

The timing diagrams that follow show the timing for external device accesses.

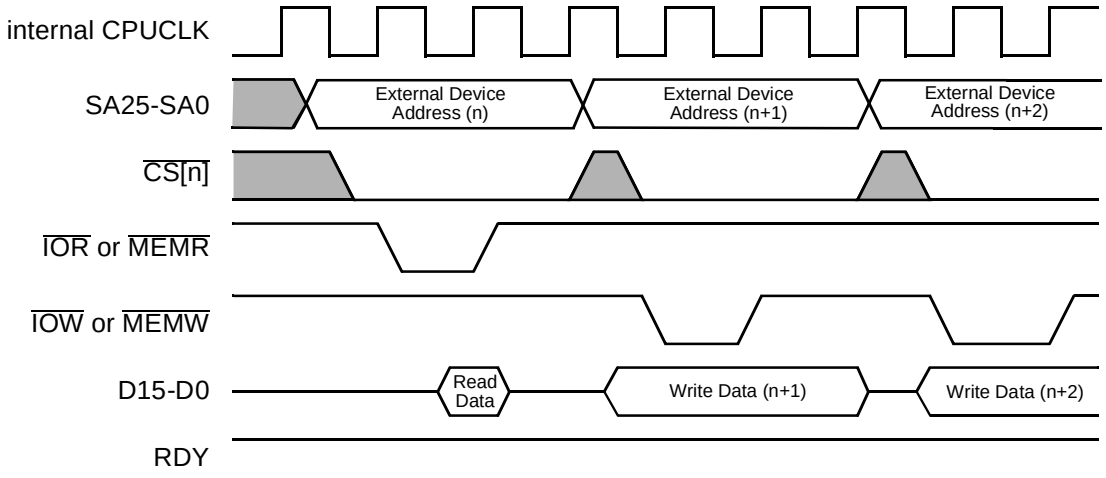


Figure 4-6 No Wait States, No Command Delays, External Interface Bus Cycles (Internal access timing is similar)

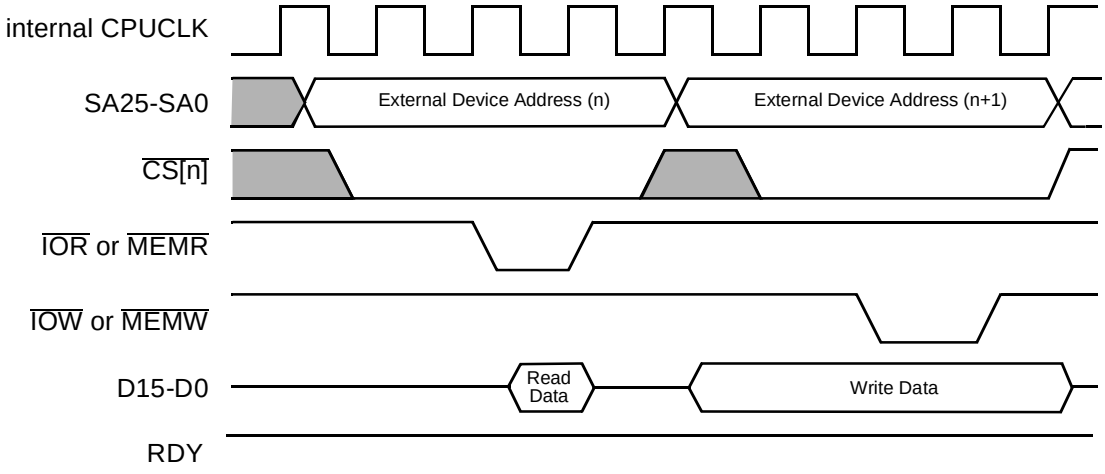


Figure 4-7 No Wait States, One Command Delay, External Bus Cycles

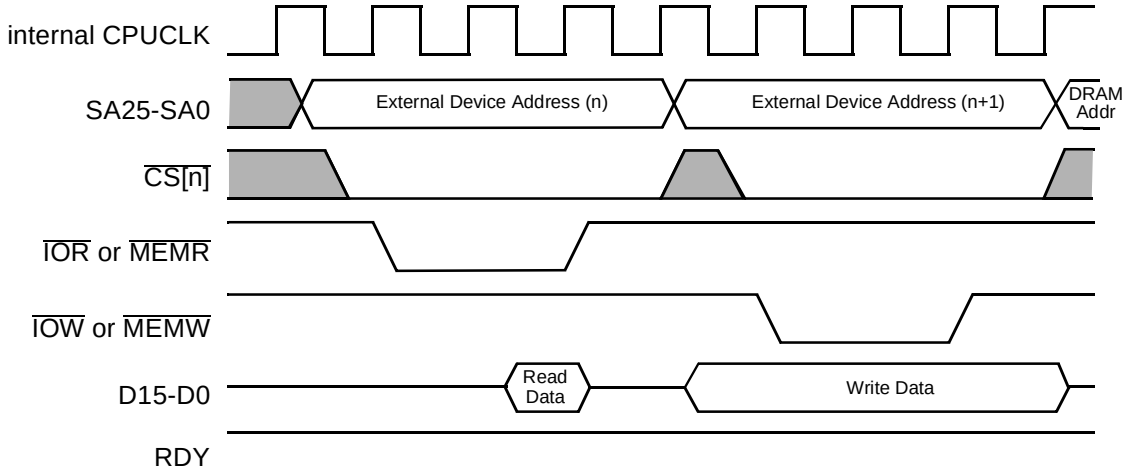


Figure 4-8 One Wait State, No Command Delays, External Interface Bus Cycles

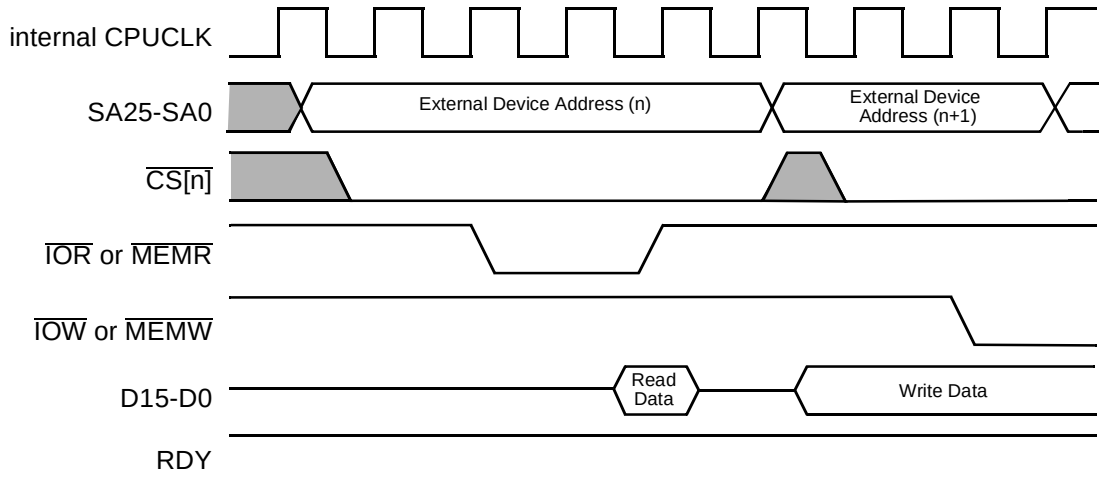


Figure 4-9 One Wait State, One Command Delay, External Interface Bus Cycles

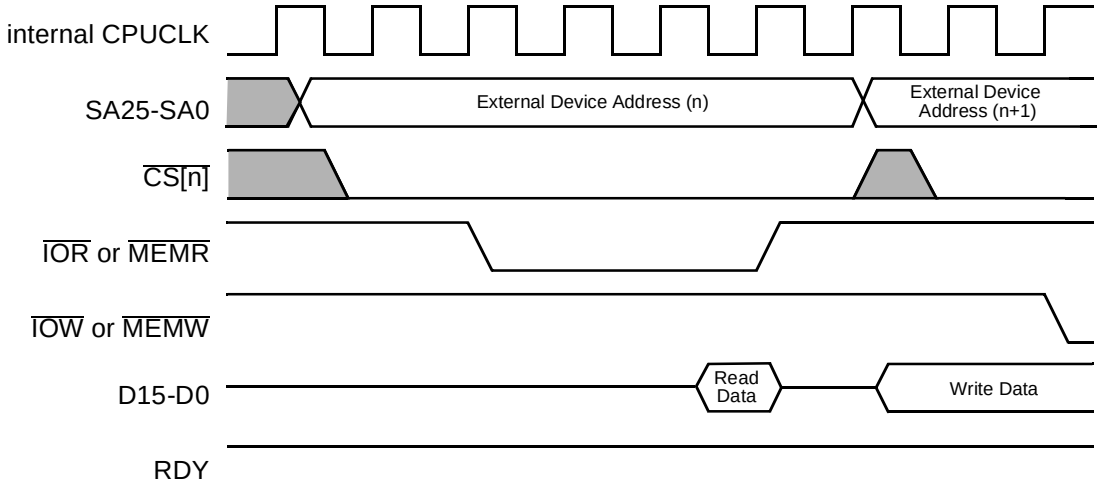


Figure 4-10 Two Wait States, One Command Delay, External Interface Bus Cycles

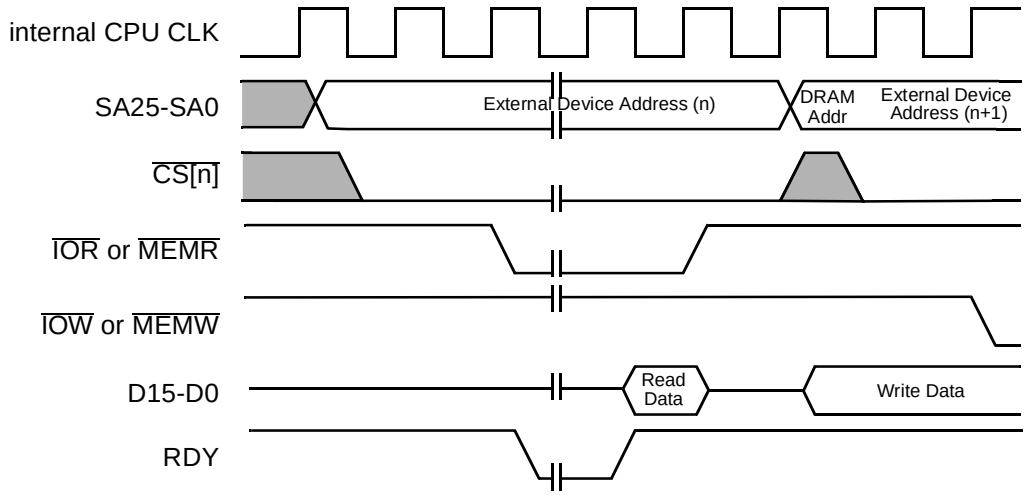


Figure 4-11 One Command Delay, Ext. Device Inserts Wait State, External Interface Bus Cycles

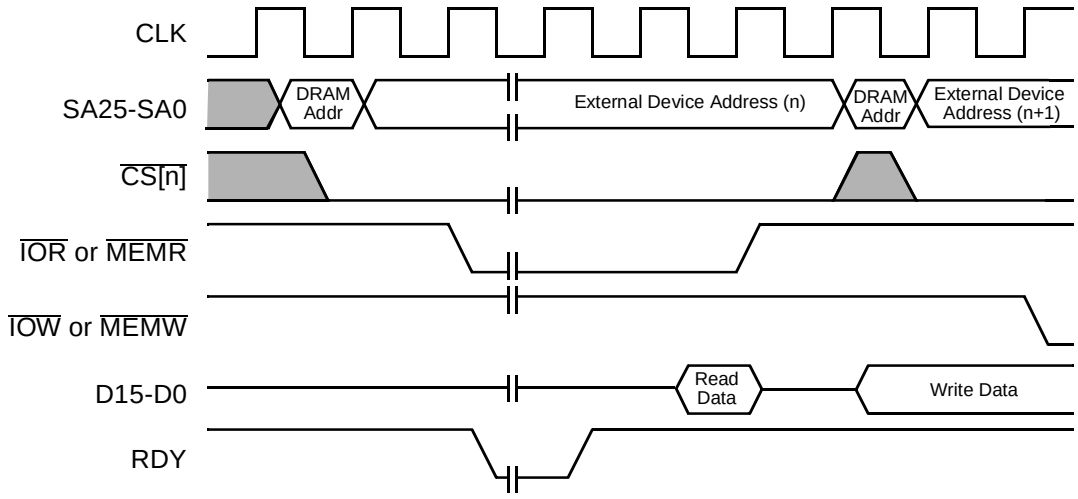


Figure 4-12 One Command Delay, Ext. Device Inserts Wait State, One Clock Period of RDY Extension, External Interface Bus Cycles

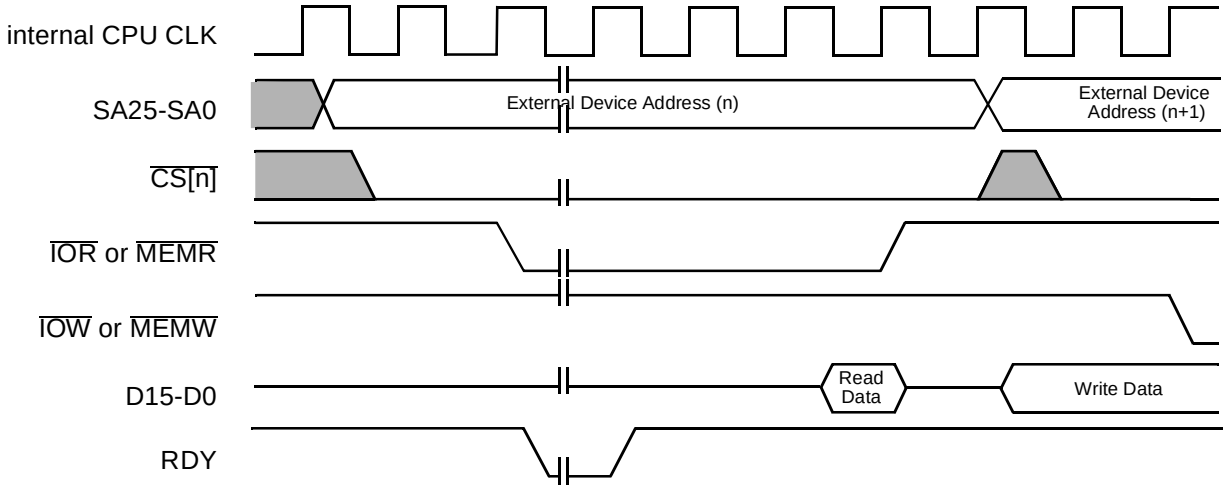
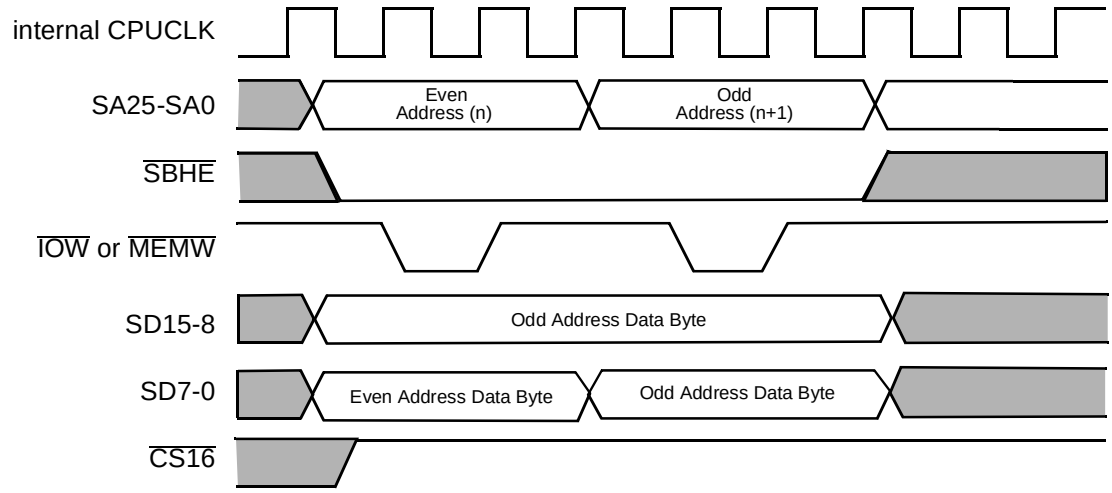
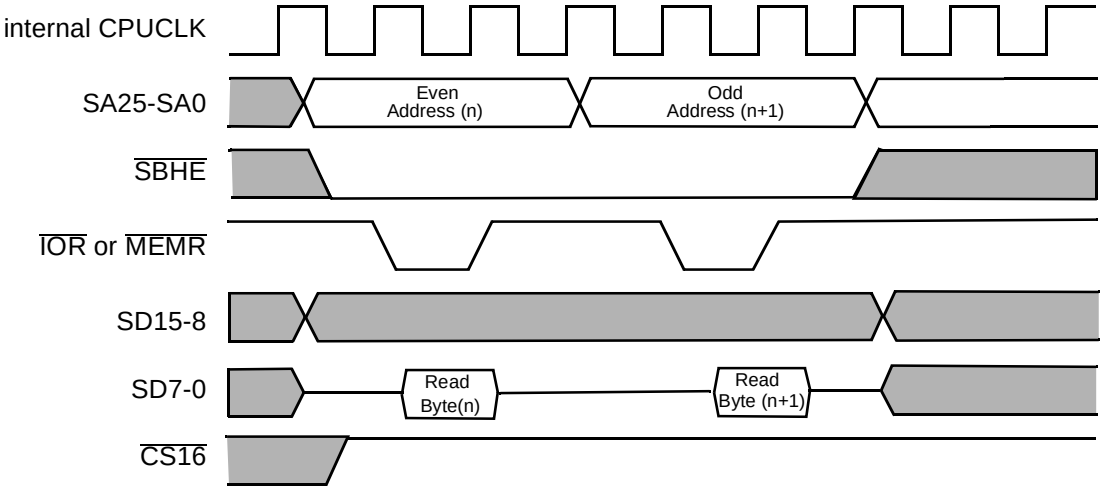


Figure 4-13 One Command Delay, Ext. Device Inserts Wait State, Two Clock Periods RDY Extension, External Interface Bus Cycles



(a) 16-bit to 8-bit Write Cycle Translation



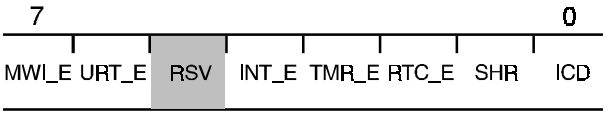
(b) 16-bit to 8-bit Read Cycle Translation

Figure 4-14 16-bit to 8-bit Cycle Translations

4.5 Bus Interface Unit Registers

4.5.1 Bus Interface Unit Control Register 1

During a system reset the bits in this register are set to 02h. This 8-bit register is located at I/O address EF00h.



I/O Map Address	Access
EF00h	R/W

Bit 7: MWI_E —MICROWIRE Interface Enable. When this bit is a one (and GPE = 1), the Bus Interface Unit will decode the CPU accesses and generate the proper access signals to the three wire interface logic. When this bit is a zero, no accesses to the three wire interface logic will be allowed. This bit is reset to zero by a system reset.

Bit 6: URT_E — UART Enable. When this bit is a one (and GPE = 1), the Bus Interface Unit will decode the CPU accesses and generate the proper access signals to the on-board UART. When this bit is a zero, no accesses to the UART will be allowed. This bit is reset to zero by a system reset.

Bit 5: Reserved

Bit 4: INT_E — Interrupt Controller Enable. When this bit is a one (and GPE = 1), the Bus Interface Unit will decode the CPU accesses and generate the proper access signals to the on-board Interrupt Controllers. When this bit is a zero, no accesses to the Interrupt Controllers will be allowed. This bit is reset to zero by a system reset.

Bit 3: TMR_E — TiMeR Enable. When this bit is a one (and GPE = 1), the Bus Interface Unit will decode the CPU accesses and generate the proper access signals to the on-board Timer. When this bit is a zero, no accesses to the Timer will be allowed. This bit is reset to zero by a system reset.

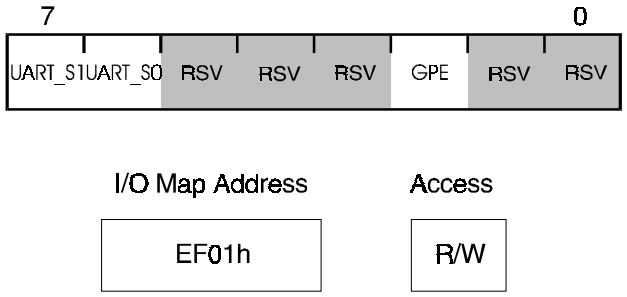
Bit 2: RTC_E — Real Time Clock access Enable. When this bit is a one (and GPE = 1), the Bus Interface Unit will decode the CPU accesses and generate the proper access signals to the on-board RTC. When this bit is a zero, no accesses to the RTC will be allowed. This bit is reset to zero by a system reset.

Bit 1: SHR — SHutdown Reset. When this bit is a one, a CPU reset will be generated when the Bus Interface Unit decodes a Shutdown cycle by the CPU. When this bit is zero, no CPU reset is generated when a Shutdown cycle is decoded. This bit is set to one by a system reset.

Bit 0: ICD — Internal Cycle Debug. When this bit is a one, an external bus cycle will be generated at the same time the internal bus cycle is performed. When this bit is a zero, there will be no external indication of internal bus cycles. This bit is set to zero by a system reset.

4.5.2 Bus Interface Unit Control Register 2

During a system reset the bits in this register are set to 00h. This 8-bit register is located at I/O address EF01h.



Bits 7-6: UART_S1 and UART_S0 — UART IO address Selection bits 1-0. These two bits determine the I/O address range associated with the internal UART. These bits are only valid when both the UART Enable bit (Bit 6 of the Bus Interface Unit Control Register 1) and the Global Peripheral Enable bit are both set to one.

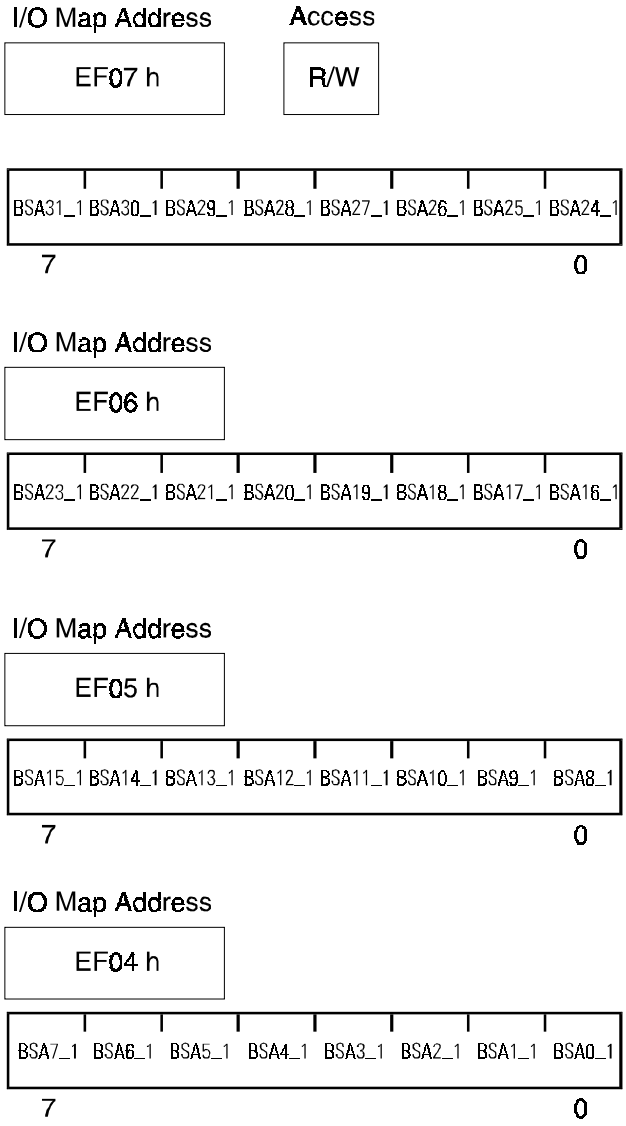
UART_S1	UART_S0	I/O Address Range
0	0	03F8h - 03FFh (COM1)
0	1	02F8h - 02FFh (COM2)
1	0	03E8h - 03EFh (COM3)
1	1	02E8h - 02EFh (COM4)

Bits 5,4,3: Reserved.

Bit 2: GPE — Global Peripheral Enable. When this bit is a zero, all accesses to peripherals will be disabled. When this bit is a one, the enable bit associated with each peripheral will determine if the peripheral may be accessed or not.

Bit 1,0: Reserved.

4.5.3 Chip Select Base Address Registers



Bits 31-0:BSA31_1- BSA0_1 — These thirty-two bits determine the address used to provide support for one of eight logical chip select decodes.

4.5.3.1 Chip Select Base Address Register 1

During a system reset the bits in these registers are not set. These four 8-bit registers are located at I/O addresses EF04h - EF07h, and may be accessed one at a time or two at a time.

4.5.3.2 Chip Select Base Address Register 2

During a system reset the bits in these registers are not set. These four 8-bit registers are located at I/O addresses EF08h - EF0Bh, and may be accessed one at a time or two at a time. These registers function like Chip Select Base Address Register 1.

4.5.3.3 Chip Select Base Address Register 3

During a system reset the bits in these registers are not set. These four 8-bit registers are located at I/O addresses EF0Ch - EF0Fh, and may be accessed one at a time or two at a time. These registers function like Chip Select Base Address Register 1.

4.5.3.4 Chip Select Base Address Register 4

During a system reset the bits in these registers are not set. These four 8-bit registers are located at I/O addresses EF10h - EF13h, and may be accessed one at a time or two at a time. These registers function like Chip Select Base Address Register 1.

4.5.3.5 Chip Select Base Address Register 5

During a system reset the bits in these registers are not set. These four 8-bit registers are located at I/O addresses EF14h - EF17h, and may be accessed one at a time or two at a time. These registers function like Chip Select Base Address Register 1.

4.5.3.6 Chip Select Base Address Register 6

During a system reset the bits in these registers are not set. These four 8-bit registers are located at I/O addresses EF18h - EF1Bh, and may be accessed one at a time or two at a time. These registers function like Chip Select Base Address Register 1.

4.5.3.7 Chip Select Base Address Register 7

During a system reset the bits in these registers are not set. These four 8-bit registers are located at I/O addresses EF1Ch - EF1Fh, and may be accessed one at a time or two at a time. These registers function like Chip Select Base Address Register 1.

4.5.3.8 Chip Select Base Address Register 8

During a system reset the bits in these registers are not set. These four 8-bit registers are located at I/O addresses EF20h - EF23h, and may be accessed one at a time or two at a time. These registers function like Chip Select Base Address Register 1.

4.5.4 Chip Select Address Mask Registers

Bits 25-0:AM25_1-AM0_1 — These twenty-six bits indicate the size or range of addresses associated with Logical Chip Select 1. The user indicates which address bits are “don’t cares” by writing a 1 to the corresponding bit(s) in this register.

Only the lower twenty-six address bits are driven off-chip (i.e. SA25-0), so no address range greater than 64 MBytes may be programmed because external devices could not dis-

tinguish which address is actually being accessed. This means that the six most significant address bits in the Chip Select Base Address Register 1 must match the corresponding CPU generated address bits for Logical Chip Select 1 to go active. Likewise this is the reason the upper six Address Mask bits are unnecessary.

4.5.4.1 Chip Select Address Mask Register 1

During a system reset the bits in these registers are not set. These four 8-bit registers are located at I/O addresses EF24h - EF27h, and are used as a mask for the associated Chip Select Address.

4.5.4.2 Chip Select Address Mask Register 2

During a system reset the bits in these registers are not set. These four 8-bit registers are located at I/O addresses EF28h - EF2Bh, and are used as a mask for the associated Chip Select Address. These registers function like Chip Select Address Mask Register 1.

4.5.4.3 Chip Select Address Mask Register 3

During a system reset the bits in these registers are not set. These four 8-bit registers are located at I/O addresses EF2Ch - EF2Fh, and are used as a mask for the associated Chip Select Address. These registers function like Chip Select Address Mask Register 1.

4.5.4.4 Chip Select Address Mask Register 4

During a system reset the bits in these registers are not set. These four 8-bit registers are located at I/O addresses EF30h - EF33h, and are used as a mask for the associated Chip Select Address. These registers function like Chip Select Address Mask Register 1.

4.5.4.5 Chip Select Address Mask Register 5

During a system reset the bits in these registers are not set. These four 8-bit registers are located at I/O addresses EF34h - EF37h, and are used as a mask for the associated Chip Select Address. These registers function like Chip Select Address Mask Register 1.

4.5.4.6 Chip Select Address Mask Register 6

During a system reset the bits in these registers are not set. These four 8-bit registers are located at I/O addresses EF38h - EF3Bh, and are used as a mask for the associated Chip Select Address. These registers function like Chip Select Address Mask Register 1.

4.5.4.7 Chip Select Address Mask Register 7

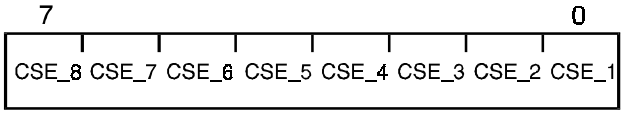
During a system reset the bits in these registers are not set. These four 8-bit registers are located at I/O addresses EF3Ch - EF3Fh, and are used as a mask for the associated Chip Select Address. These registers function like Chip Select Address Mask Register 1.

4.5.4.8 Chip Select Address Mask Register 8

During a system reset the bits in these registers are not set. These four 8-bit registers are located at I/O addresses EF40h - EF43h, and are used as a mask for the associated Chip Select Address. These registers function like Chip Select Address Mask Register 1.

4.5.5 Chip Select Enable Register

During a system reset the bits in this register are reset to zero. This 8-bit register is located at I/O addresses EF02h and contains the enables for each of the eight Logical Chip Select decodes.

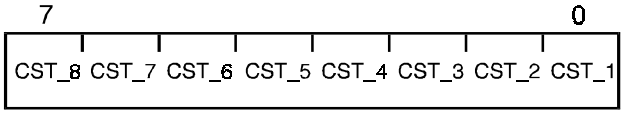


I/O Map Address	Access
EF02h	R/W

Bits 7-0: CSE_8-CSE_1 — These eight bits enable or disable (1 or 0, respectively) the associated Logical Chip Select decodes. Before programming or changing either the Chip Select Base Address or Address Range registers one should always disable the Logical Chip Select decode by writing a zero to the appropriate bit(s).

4.5.6 Chip Select Type Register

During a system reset the bits in this register are not set. This 8-bit register is located at I/O addresses EF03h and contains the type of cycle (memory or IO) for each of the eight Chip Select decodes.



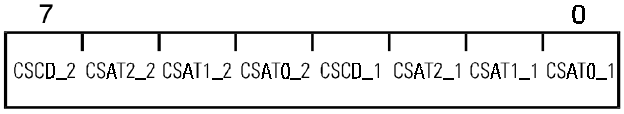
I/O Map Address	Access
EF03h	R/W

Bits 7-0:CST_8-CST_1 — These eight bits determine the type of cycle (memory when a 1 and IO when a 0) associated with each Chip Select decode. **Before programming or changing either the Chip Select Type registers one should always disable the Chip Select decode by writing a zero to the Chip Select Enable bit appropriate bit(s)**.

4.5.7 Chip Select Access Time Registers

4.5.7.1 Chip Select Access Time Register 1

During a system reset the bits in this register are all set to ones. This 8-bit register is located at I/O address EF50h and contains the access timing information for Logical Chip Select decodes 1 and 2.



I/O Map Address	Access
EF50h	R/W

Bit 7: CSCD_2 — Chip Select 2 Command Delay. When this bit is a one, an extra clock period of address setup time will inserted into the accesses associated with this Logical Chip Select decode.

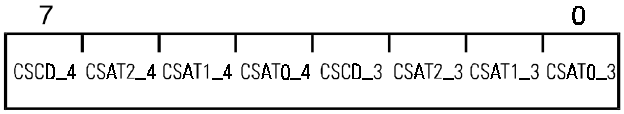
Bits 6-4: CSAT2_2-CSAT0_2 — Chip Select 2 Access Time. These three bits determine the number of clock cycles the external bus interface command strobes will be programmed to be active low. The number of clock cycles the command strobes are active will always be one greater than the value in these bits.

Bit 3: CSCD_1 — Chip Select 1 Command Delay. When this bit is a one, an extra clock period of address setup time will inserted into the accesses associated with this Logical Chip Select decode.

Bits 2-0: CSAT2_1-CSAT0_1 — Chip Select 1 Access Time. These three bits determine the number of clock cycles the external bus interface command strobes will be programmed to be active low. The number of clock cycles the command strobes are active will always be one greater than the value in these bits.

4.5.7.2 Chip Select Access Time Register 2

During a system reset the bits in this register are all set to ones. This 8-bit register is located at I/O address EF51h and contains the access timing information for Logical Chip Select decodes 3 and 4.



I/O Map Address	Access
EF51h	R/W

Bit 7: CSCD_4 — Chip Select 4 Command Delay. When this bit is a one, an extra clock period of address setup time will inserted into the accesses associated with this Logical Chip Select decode.

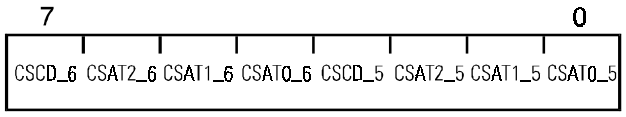
Bits 6-4: CSAT2_4 — Chip Select 4 Access Time. These three bits determine the number of clock cycles the external bus interface command strobes will be programmed to be active low. The number of clock cycles the command strobes are active will always be one greater than the value in these bits.

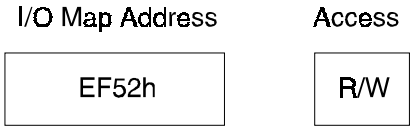
Bit 3: CSCD_3 — Chip Select 3 Command Delay. When this bit is a one, an extra clock period of address setup time will inserted into the accesses associated with this Logical Chip Select decode.

Bits 2-0: CSAT2_3-CSAT0-3 — Chip Select 3 Access Time. These three bits determine the number of clock cycles the external bus interface command strobes will be programmed to be active low. The number of clock cycles the command strobes are active will always be one greater than the value in these bits.

4.5.7.3 Chip Select Access Time Register 3

During a system reset the bits in this register are all set to ones. This 8-bit register is located at I/O address EF52h and contains the access timing information for Chip Select decodes 5 and 6.





Bit 7: CSCD_6 — Chip Select 6 Command Delay. When this bit is a one, an extra clock period of address setup time will inserted into the accesses associated with this Logical Chip Select decode.

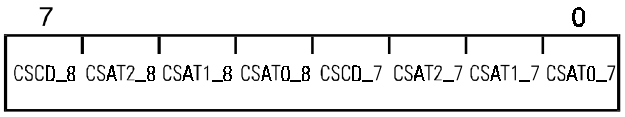
Bits 6-4: CSAT2_6-CSAT0_6 — Chip Select 6 Access Time. These three bits determine the number of clock cycles the external bus interface command strobes will be programmed to be active low. The number of clock cycles the command strobes are active will always be one greater than the value in these bits.

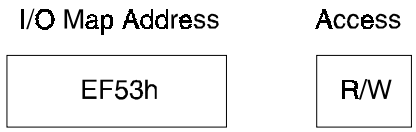
Bit 3: CSCD_5 — Chip Select 5 Command Delay. When this bit is a one, an extra clock period of address setup time will inserted into the accesses associated with this Logical Chip Select decode.

Bits 2-0: CSAT2_5-CSAT0_5 — Chip Select 5 Access Time. These three bits determine the number of clock cycles the external bus interface command strobes will be programmed to be active low. The number of clock cycles the command strobes are active will always be one greater than the value in these bits.

4.5.7.4 Chip Select Access Time Register 4

During a system reset the bits in this register are all set to ones. This 8-bit register is located at I/O address EF53h and contains the access timing information for Logical Chip Select decodes 7 and 8.





Bit 7: CSCD_8 — Chip Select 8 Command Delay. When this bit is a one, an extra clock period of address setup time will inserted into the accesses associated with this Logical Chip Select decode.

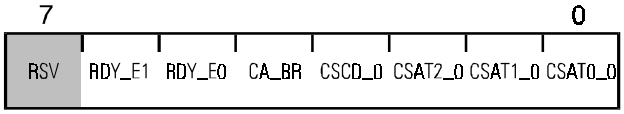
Bits 6-4: CSAT2_8-CSAT0_8 — Chip Select 8 Access Time. These three bits determine the number of clock cycles the external bus interface command strobes will be programmed to be active low. The number of clock cycles the command strobes are active will always be one greater than the value in these bits.

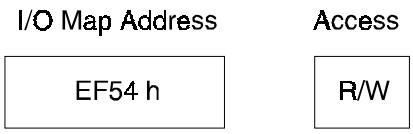
Bit 3: CSCD_7 — Chip Select 7 Command Delay. When this bit is a one, an extra clock period of address setup time will inserted into the accesses associated with this Logical Chip Select decode.

Bits 2-0: CSAT2_7-CSAT0_7 — Chip Select 7 Access Time. These three bits determine the number of clock cycles the external bus interface command strobes will be programmed to be active low. The number of clock cycles the command strobes are active will always be one greater than the value in these bits.

4.5.7.5 Boot ROM Access Time Register

During a system reset this register is set to 6Fh. This 7-bit register is located at I/O address EF54h and contains the access timing information for Boot ROM Chip Select 0($\overline{CS[0]}$). Also bits 5 and 6 of this register determine the minimum number of CPUCLK periods following the assertion of RDY, that the associated command strobe (i.e., $\overline{IOR}$, $\overline{IOW}$, $\overline{MEMR}$, or $\overline{MEMW}$) may go inactive high to terminate an access. The default number is two CPUCLK periods.





Bits 7: Reserved.

Bits 6-5: RDY_E1 and RDY_E0 — ReaDY Extension bits 1-0.

RDY_E1	RDY_E0	CPUCLK Periods of Extension
0	0	Zero
0	1	One
1	0	Two (reset default)
1	1	Two

Bit 4: CA_BR — Cacheable Boot ROM Chip Select. When a one is written into this bit, it indicates that the Boot ROM address range (FFFF_0000h - FFFF_FFFFh) is cacheable. When this bit is a zero, this address range is uncacheable.

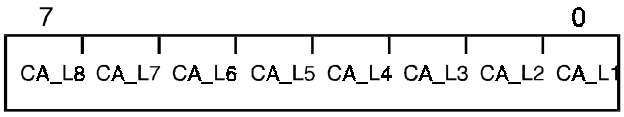
Bit 3: CSCD_0 — Chip Select 0 Command Delay. When this bit is a one, an extra clock period of address setup time will inserted into the accesses to the Boot ROM address range (FFFF_0000h - FFFF_FFFF).

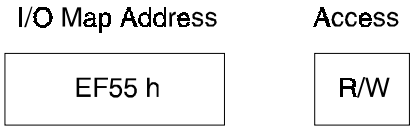
Bits 2-0: CSAT2_0-CSAT0_0 — Chip Select 0 Access Time. These three bits determine the number of clock cycles the external bus interface command strobes will be programmed to be active low. The number of clock cycles the command strobes are active will always be one greater than the value in these bits.

4.5.8 Cacheable Chip Selects Register

During a system reset the bits in this register are set to 00h. This 8-bit register is located at I/O address EF55h.

Writing a 1 into a bit in this register indicates that the corresponding Logical Chip Select decodes a cacheable memory range. The **NS486SXL** CPU will only allow the caching of instruction code fetched from cacheable memory ranges.



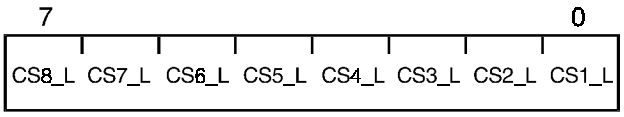


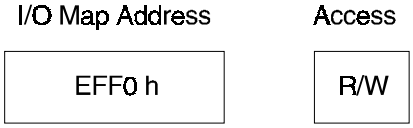
Bit 7-0: CA_L8 - CA_L1 — CacheAble Logical Chip Select 8-1. When a 1 is written into a bit in this register, it indicates that the corresponding Logical Chip Select decodes a cacheable memory range.

4.5.9 External Chip Selection Registers

4.5.9.1 Chip Select Local Device Register

This register determines the location of the respective Chip Select relative to the 74x245 buffers. If the respective bit is set to a 0, then the device using the chip select is on the other side (relative to the NS486SXL) of the buffers. A 1 set in the relevant bit will indicate that the device using that chip select is on the near side of the buffers. At reset, this registers contents are all zero.



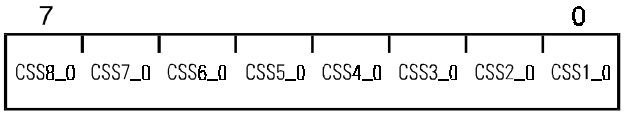


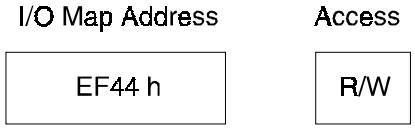
Bits 7-0: CS8_L1-CS1_L — Chip Select Local Device. When this bit is a 1, the device that uses the related Chip Select is located local to the NS486SXL pins. If this bit is set to a 0, then teh device that uses that Chip Select is shown to be on the other side of the 74x245 buffers, relative to the NS486SXL.

4.5.9.2 Boot ROM Selection Register

During a system reset the bits in this register are all set to zeros. This 8-bit register is located at I/O address EF44h and contains selection bits for determining which Logical Chip Selects will be associated with the External Chip Select 0 ($\overline{CS[0]}$).

NOTE: $\overline{CS0}$ will always go active for memory accesses to the most significant 64 Kbytes of memory (FFFF0000h - FFFFFFFFh). This register allows the user to specify additional memory range(s) that will cause $\overline{CS[0]}$ to go active low.

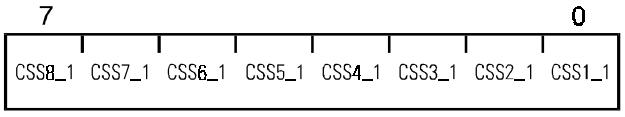


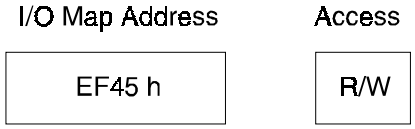


Bits 7-0: CSS8_0- CSS1_0 — Chip Select Selection. When a bit is a one in this register, it indicates that when there is an active Logical Chip Select associated with that decode logic, the external Chip Select 0 ($\overline{CS[0]}$) signal will go active.

4.5.9.3 External Chip Select Selection Register 1

During a system reset the bits in this register are all set to zeros. This 8-bit register is located at I/O address EF45h and contains selection bits for determining which Logical Chip Selects will be associated with the External Chip Select 1 ($\overline{CS[1]}$).

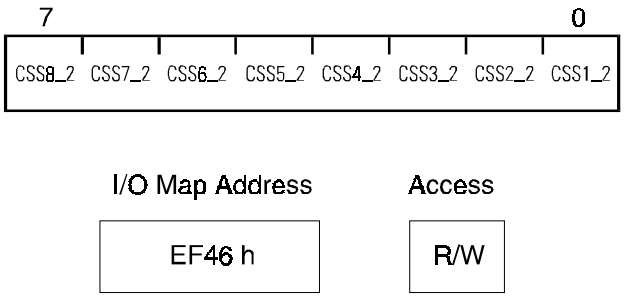




Bits 7-0: CSS8_1-CSS1_1 — Chip Select Selection. When a bit is a one in this register, it indicates that when there is an active Logical Chip Select associated with that decode logic, the external Chip Select 1 ($\overline{CS[1]}$) signal will go active.

4.5.9.4 External Chip Select Selection Register 2

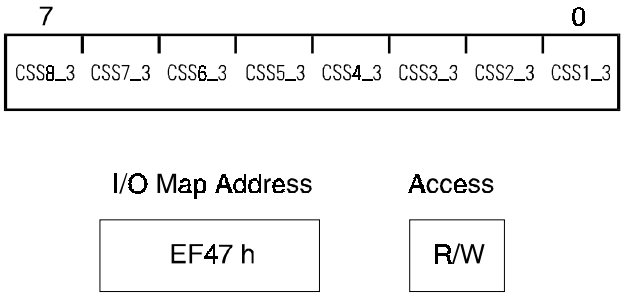
During a system reset the bits in this register are all set to zeros. This 8-bit register is located at I/O address EF46h and contains selection bits for determining which Logical Chip Select decodes will be associated with the External Chip Select 2 ($\overline{CS}$[2]).



Bits 7-0: CSS8_2- CSS1_2 — Chip Select Selection. When a bit is a one in this register, it indicates that when there is an active Logical Chip Select associated with that decode logic, the external Chip Select 2 ($\overline{CS}$[2]) signal will go active.

4.5.9.5 External Chip Select Selection Register 3

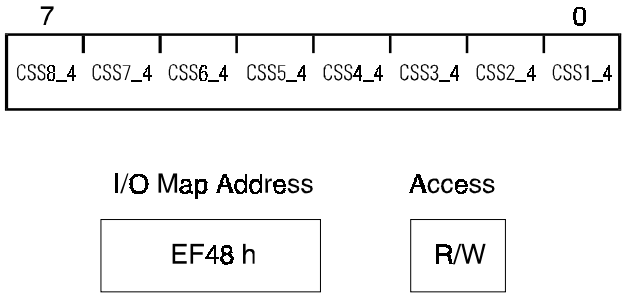
During a system reset the bits in this register are all set to zeros. This 8-bit register is located at I/O address EF47h and contains selection bits for determining which Logical Chip Select decodes will be associated with the External Chip Select 3 ($\overline{CS}$[3]).



Bits 7-0: CSS8_3 - CSS1_3 — Chip Select Selection. When a bit is a one in this register, it indicates that when there is an active Logical Chip Select associated with that decode logic, the external Chip Select 3 ($\overline{CS}$[3]) signal will go active.

4.5.9.6 External Chip Select Selection Register 4

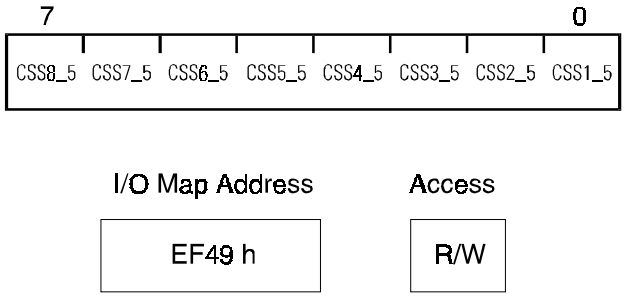
During a system reset the bits in this register are all set to zeros. This 8-bit register is located at I/O address EF48h and contains selection bits for determining which Logical Chip Select decodes will be associated with the External Chip Select 4 ($\overline{CS}$[4]).



Bits 7-0: CSS8_4 - CSS1_4 — Chip Select Selection. When a bit is a one in this register, it indicates that when there is an active Logical Chip Select associated with that decode logic, the external Chip Select 4 ($\overline{CS}$[4]) signal will go active.

4.5.9.7 External Chip Select Selection Register 5

During a system reset the bits in this register are all set to zeros. This 8-bit register is located at I/O address EF49h and contains selection bits for determining which Logical Chip Select decodes will be associated with the External Chip Select 5 ($\overline{CS}$[5]).

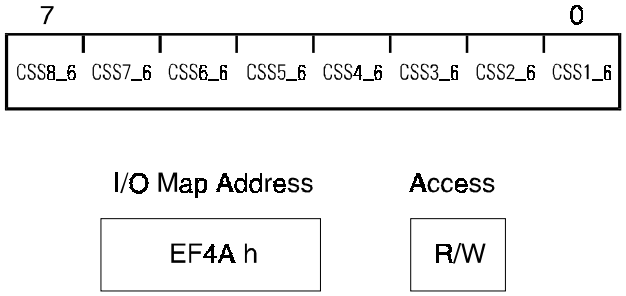


Bits 7-0: CSS8_5 - CSS1_5 — Chip Select Selection. When a bit is a one in this register,

it indicates that when there is an active Logical Chip Select associated with that decode logic, the External Chip Select 5 ($\overline{CS[5]}$) signal will go active.

4.5.9.8 External Chip Select Selection Register 6

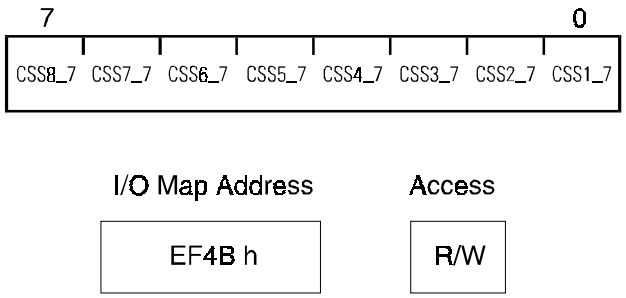
During a system reset the bits in this register are all set to zeros. This 8-bit register is located at I/O address EF4Ah and contains selection bits for determining which Logical Chip Select decodes will be associated with the External Chip Select 6 ($\overline{CS[6]}$).



Bits 7-0: CSS8_6 - CSS1_6 — Chip Select Selection. When a bit is a one in this register, it indicates that when there is an active Logical Chip Select associated with that decode logic, the external Chip Selcet6 ($\overline{CS[6]}$) signal will go active.

4.5.9.9 External Chip Select Selection Register 7

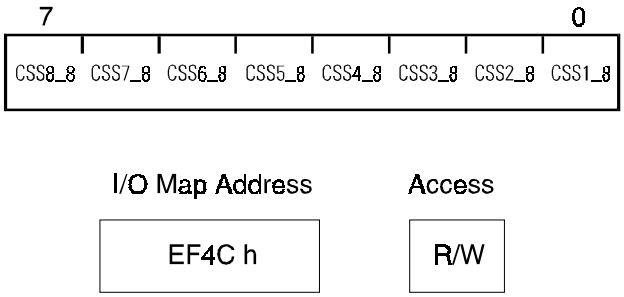
During a system reset the bits in this register are all set to zeros. This 8-bit register is located at I/O address EF4Bh and contains selection bits for determining which Logical Chip Select decodes will be associated with the External Chip Select 7 ($\overline{CS[7]}$).



Bits 7-0: CSS8_7 - CSS1_7 — Chip Select Selection. When a bit is a one in this register, it indicates that when there is an active Logical Chip Select associated with that decode logic, the external Chip Select 7 ($\overline{CS[7]}$) signal will go active.

4.5.9.10 External Chip Select Selection Register 8

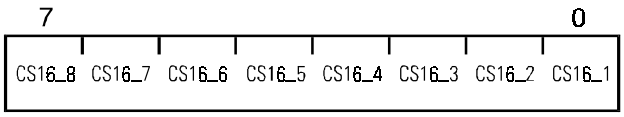
During a system reset the bits in this register are all set to zeros. This 8-bit register is located at I/O address EF4Ch and contains selection bits for determining which Logical Chip Select decodes will be associated with the External Chip Select 8 ($\overline{CS[8]}$).



Bits 7-0: CSS8_8 - CSS1_8 — Chip Select Selection. When a bit is a one in this register, it indicates that when there is an active Logical Chip Select associated with that decode logic, the external Chip Selcet8 ($\overline{CS[8]}$) signal will go active.

4.5.10 Programmable 16-bit Logical Chip Select Register

During a system reset the bits in this register are all set to zeros. This 8-bit register is located at I/O address EF57h. When a bit in this register is set to a one, its associated Logical Chip Select access will be treated as a 16-bit access, regardless of the state of the $\overline{CS16}$ signal. When a bit in this register is zero, its Logical Chip Select access will generate the appropriate bus cycles based on the $\overline{CS16}$ signal.



I/O Map Address	Access
EF57 h	R/W

Bits 7-0: CS16_8 - CS16_1 — Logical Chip Select 8-1, 16-bit device. Bit 7 is associated with logical chip select 8, bit 6 with logical chip select 7, and so on. When set to 1, an access to the selected logical chip select will be treated as a 16-bit access, regardless of the status of $\overline{CS16}$. When set to 0,an access to the logical chip select will generate the appropriate bus cycles based upon the $\overline{CS16}$ signal.

4.6 Auxiliary Processor, Shared Memory Interface

NS486SXL supports the use of an Auxiliary Processor, with a sharing of memory space. The Auxiliary Processor interface provides a low cost interface for sharing memory belonging to an Auxiliary Processor or any other processor including another **NS486SXL**. Only a small amount of additional logic, along with a set of TRI-STATE buffers for address, data and control signals, is required.

4.6.1 Auxiliary Processor Communications

There are two methods of transferring information to/from shared memory with the **NS486SXL**; the first is under CPU control and the second is via the DMA Controller.

Note: The Auxiliary Processor Communications interface supports only memory mapped shared memory; IO mapping of the shared memory at present is not supported.

4.6.2 CPU Controlled Transfers

When under CPU control, one or more of the programmable Logical Chip Selects should be programmed to indicate that they are to be used as Auxiliary Processor Communication chip select(s). **Furthermore, a command delay must be programmed for any Logical Chip Select(s) used to support the Auxiliary Processor shared memory space**. (The command delay requirement is made to allow the use of more efficient arbitration logic within the **NS486SXL**.) The Auxiliary Processor Chip Select Selection Register (located at I/O address EF56h) determines which Logical Chip Selects support Auxiliary Processor shared memory.

After the appropriate Logical Chip Select(s) have been programmed, any CPU access to the address range associated with given Logical Chip Select(s) will generate a External Bus Request ($\overline{EREQ}$). Until an External Bus Acknowledge ($\overline{EACK}$) is received in reply the **NS486SXL's** bus will loop in a command delay state. Once $\overline{EACK}$ is driven low, the combination of both $\overline{EREQ}$ and $\overline{EACK}$ both being low will result in the Shared Memory Drive Control signal ($\overline{DRV}$) going active low. After one more CPUCLK period command delay, the memory strobe ($\overline{MEMR}$ or $\overline{MEMW}$) will be driven low for its programmed period of time, plus any RDY inserted wait states. After the rising edge of the

$\overline{\text{MEMR}}$ or $\overline{\text{MEMW}}$ strobe, one CPUCLK period of hold time will be associated with the access; after which $\overline{\text{EREQ}}$ (and thus $\overline{\text{DRV}}$) will be taken inactive high.

The exception to the above description is when a **NS486SXL** 16-bit CPU access to an 8-bit shared memory is performed, or when the CPU performed a locked bus cycle sequence. In such cases the $\overline{\text{EREQ}}$ signal will stay active low until the transfer sequence is completed and will only return inactive high at the end of the transfer cycle.

It should be noted that in general, this method of accessing shared memory only allows a single transfer with the shared memory by the **NS486SXL's** CPU before it releases the shared memory by taking $\overline{\text{EREQ}}$ inactive high. Furthermore, it should be noted that any back to back accesses to the shared memory by the **NS486SXL's** CPU must take into consideration the latency issue of the $\overline{\text{EREQ}}$ going inactive high to the $\overline{\text{EACK}}$ going inactive. If this period of time is greater than two **NS486SXL** CPUCLK periods, then back to back bus cycles must be avoided by software means.

The following block diagram shows a conceptual shared memory interface between the **NS486SXL** and an Auxiliary Processor. External circuitry must use the $\overline{\text{DRV}}$ and $\overline{\text{MEMR}}$ signals to determine the direction of the data buses; otherwise all of the other signals are driven from the **NS486SXL's** bus to the Auxiliary Processor's Bus.

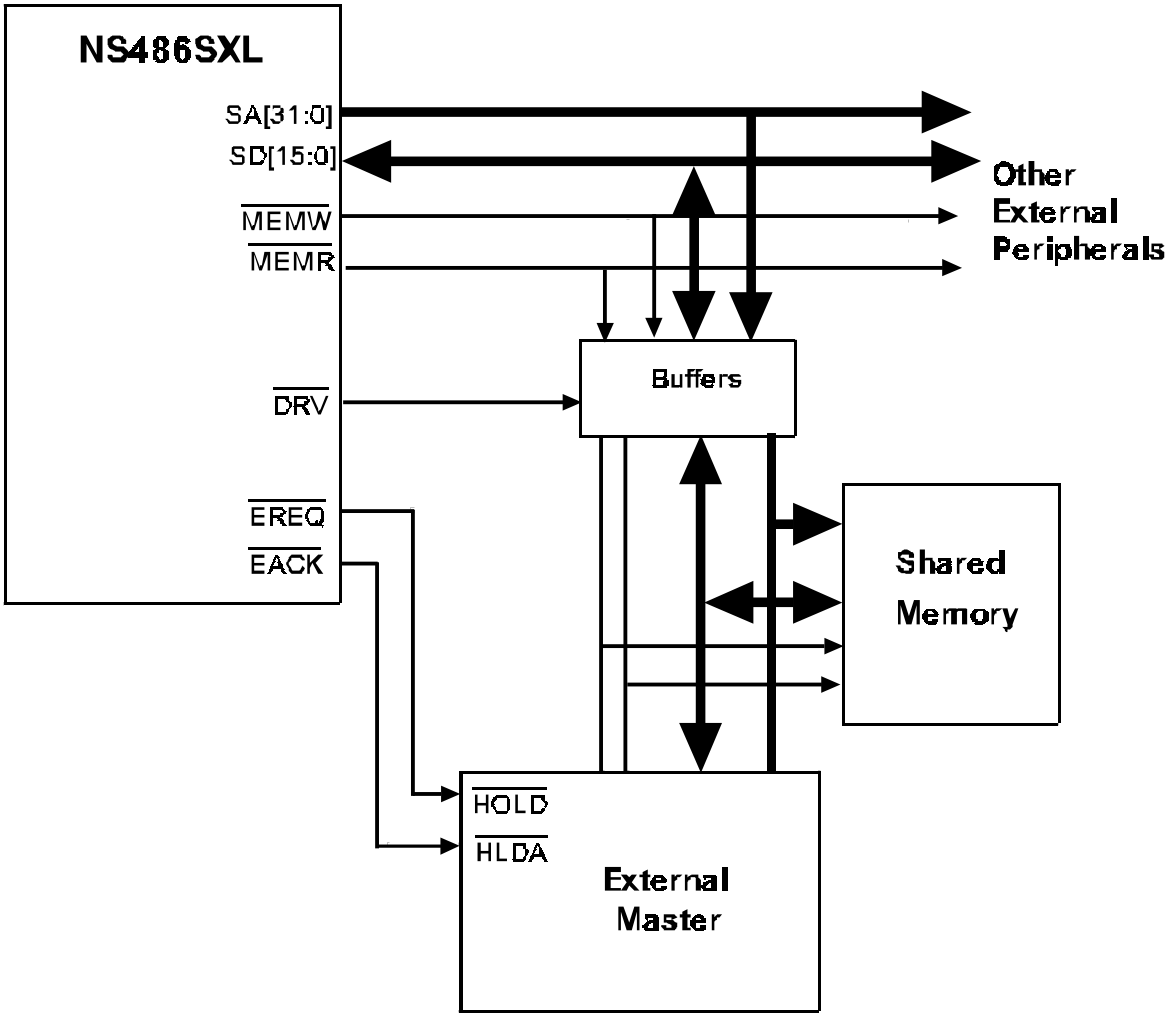
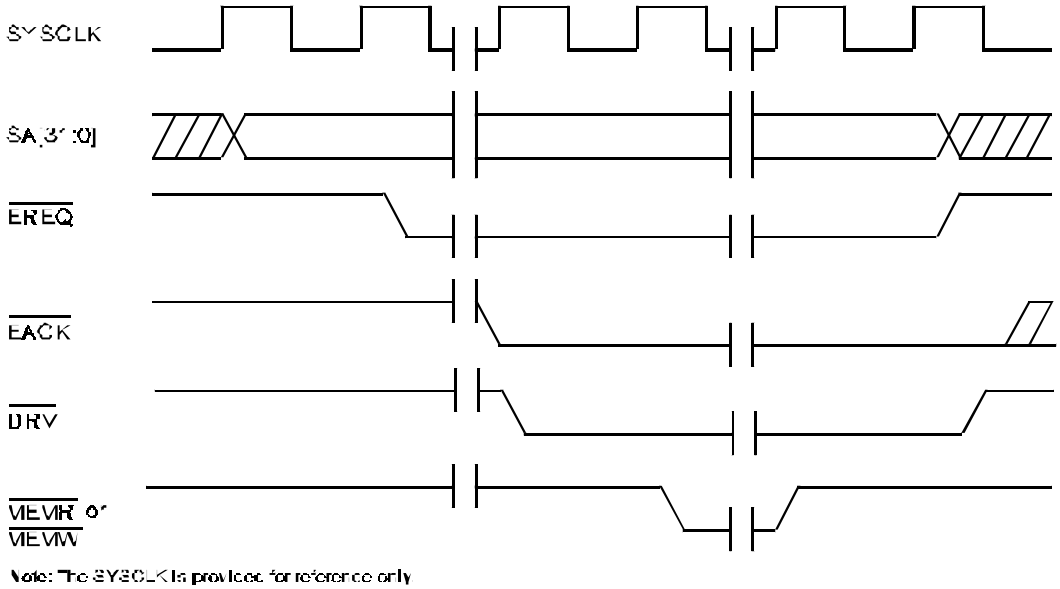


Figure 4-15 Auxiliary Processor Shared Memory Block Diagram



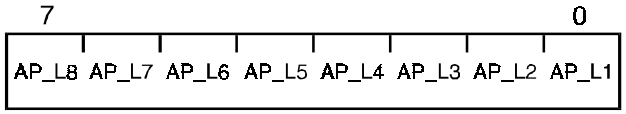
Note: The SYSCLK is provided for reference only

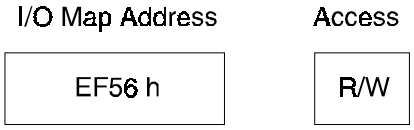
Figure 4-16 Shared Memory Single Access (CPU Controlled)

4.6.3 Auxiliary Processor Chip Select Selection Register

During a system reset the bits in this register are set to 00h. This 8-bit register is located at I/O address EF56h.

Writing a 1 into a bit in this register indicates that the corresponding Logical Chip Select decodes a Auxiliary Processor shared memory range. When an access is made to that Auxiliary Processor shared memory range the Bus Interface Unit will perform the arbitration for the Auxiliary Processor shared memory and perform the appropriate bus cycle.





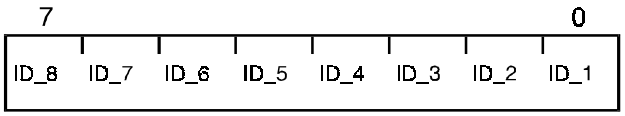
Bit 7-0: AP_L8 - AP_L1 — Auxiliary Processor Logical Chip Select 8-1. When a 1 is written into a bit in this register, it indicates that the corresponding Logical Chip Select decodes an Auxiliary Processor shared memory range.

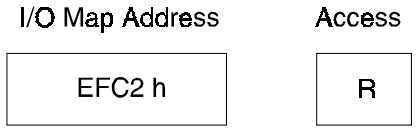
4.7 Device ID and Revision Registers

The **NS486SXL** contains two 8-bit read-only registers that contain the Part ID and Revision, respectively. These two registers allow software the ability to determine specifically determine the device type and revision.

4.7.1 Device ID Register

The **NS486SXL** device ID number is contained in the read-only Device ID Register at I/O address EFC2h. The **NS486SXL** device ID number is 02h.

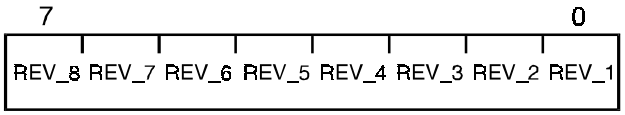


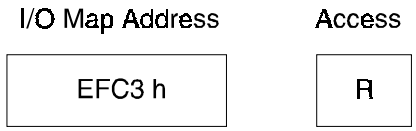


Bits 7-0: ID_7-ID_1 — These eight bits provide the device ID number of the **NS486SXL**, which is 01h.

4.7.2 Device Revision Register

The **NS486SXL** device revision number is contained in the read-only Device Revision Register at I/O address EFC3h. The present **NS486SXL** device ID number is 01h.





Bits 7-0: REV_7-REV_1 — These eight bits provide the device revision number of the **NS486SXL**, which is currently 01h.

5.0 NS486SXL Bus Master Operation

5.1 INTRODUCTION

An important new feature of the NS486SXL is the support for external bus masters.

External bus masters can access all external peripherals and memory, the 'SXL controlled DRAM and the NS486SXL internal peripherals.

Below is a block diagram showing the typical system environment that supports external bus masters on the NS486SXL bus.

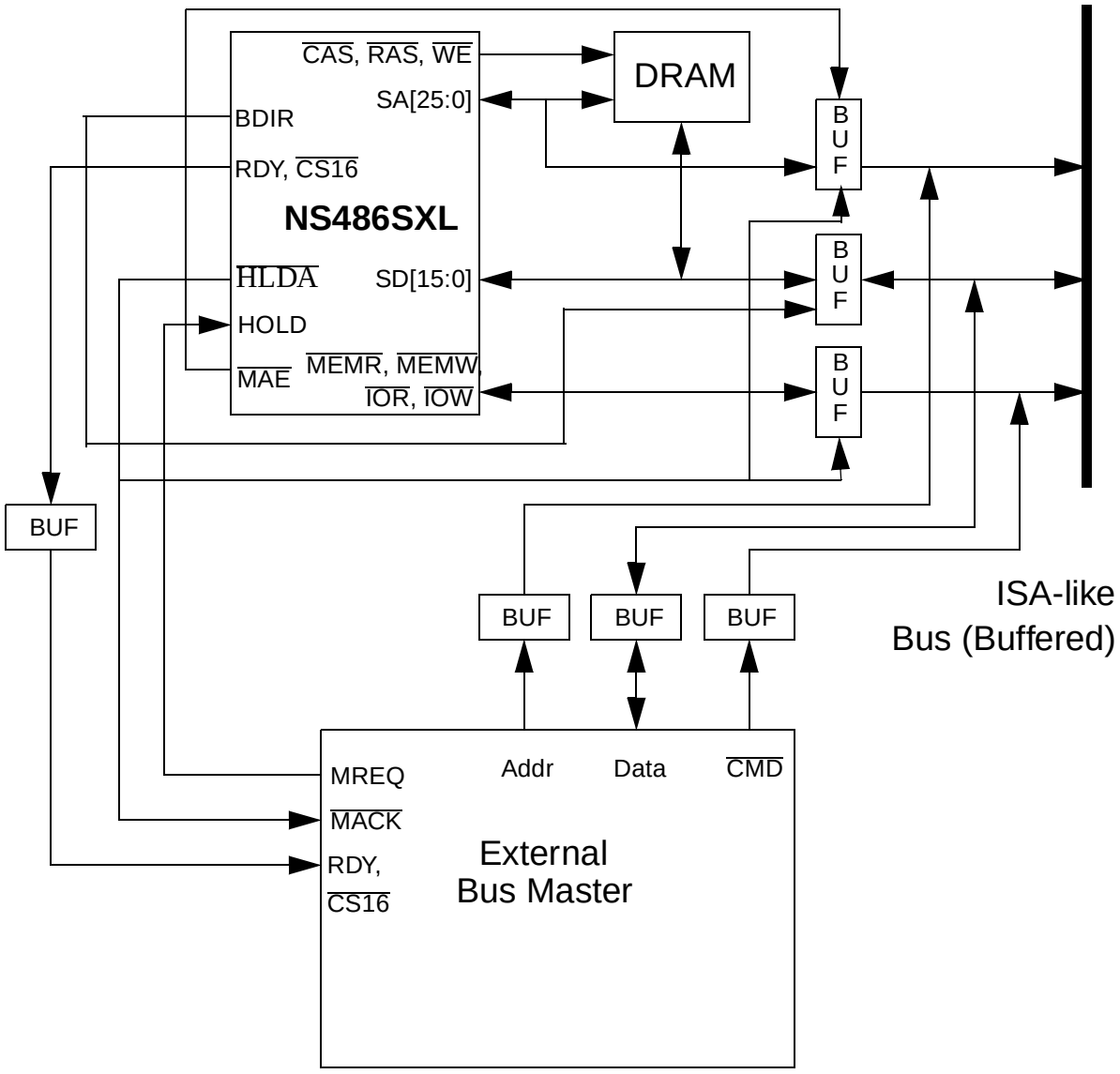


Figure 5-1 Bus Mastering System Block Diagram

The blocks in Figure 5-1, "Bus Mastering System Block Diagram," are defined as follows:

NS486SXL- This is the National Semiconductor 486 SX-Lite. The signals on the right hand side of this block are the same functions as are found in the NS486SXF (the 'SXF is described in the databook, "NS486SXF optimized 32-bit 486-class Controller With On-chip Peripherals for Embedded Systems"). The signals on the left are new pins that are provided to support Bus Mastering. These new signals will be discussed in greater detail later.

DRAM - This is the DRAM supported by the NS486SXL. Like an NS486SXF system, the DRAM resides locally on the NS486SXL's address bus (SA[12:1]) and data (SD[15:0]) bus.

Vertical BUF- The three vertical BUF blocks indicate the suggested address, data and optional command buffers for all non-DRAM in the system. The use of these buffers must be determined by the system designer and all of them may be optional in some configurations. The direction of the data buffers may be controlled via the BDIR signal, see section 5.2.4 on page 101.

The direction of the (optional) command strobe buffer and the (optional) address buffer should be controlled by the $\overline{\text{HLDA}}$ signal.

NOTE: When $\overline{\text{HLDA}}$ is a one these buffers should drive out from the NS486SXL's local bus pins. When $\overline{\text{HLDA}}$ is a zero, these buffers should drive in from the NS486SXL's ISA slots.

Every block discussed so far is the same as that of the NS486SXF. The following blocks are new ones.

External Bus Master- This block represents the External Bus Master logical function which means it can represent a single IC or multiple ICs. The requirements of this block are discussed in greater detail later, but in short it is required to generate a hold request (HOLD), receive a hold acknowledge ($\overline{\text{HLDA}}$) and produce an address, the appropriate command strobe and either drive or receive data depending on the cycle (write or read, respectively).

Horizontal BUF- These four horizontal buffers isolate the External Master's command strobes, ready feedback, chip select 16-bit, data and address from the NS486SXL ISA-like bus until the $\overline{\text{HLDA}}$ signal is active low. It is the responsibility of the External Bus Master to avoid driving the ISA-like signals whenever $\overline{\text{HLDA}}$ is inactive high.

NOTE: If there is no optional Vertical address buffer, then the External Master must guarantee that it does not drive the NS486SXL's SA[31:0] when $\overline{\text{MAE}}$ is inactive high OR when $\overline{\text{HLDA}}$ is inactive high.

5.2 Access Sequence

The following discussion and timing diagrams will walk through an External Bus Master access.

Figure 5-2, "HOLD and HLDA relationship,", below shows that the External Bus Master must drive the hold request signal (HOLD) active high; this is the request for control of the NS486SXL local bus pins. As indicated in the timing diagram, the NS486SXL will indicate that it is releasing control of the bus when it drives the hold acknowledge signal ($\overline{\text{HLDA}}$) active low.

The amount of time between HOLD going active high and the NS486SXL driving $\overline{\text{HLDA}}$ active high is not specified.

On the other hand, the de-assertion of HOLD to the de-assertion of $\overline{\text{HLDA}}$ will have a maximum value, which will be specified.

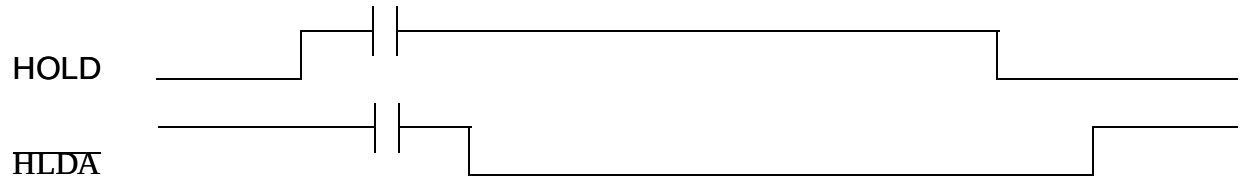


Figure 5-2 HOLD and HLDA relationship

The memory address enable signal ($\overline{MAE}$) will be active low (active-low because that is the enabling level of 74xxx244 or 74xxx245 buffers), at all times $\overline{HLDA}$ is inactive high. However, $\overline{MAE}$ will go high to TRI-STATE the address at different points during External Master transfers. The unknown pulses on the $\overline{MAE}$ signal (in Figure 5-3, "MAE Example,") indicate possible high pulses.

Nonetheless, $\overline{MAE}$ will only be inactive while $\overline{HLDA}$ is asserted.

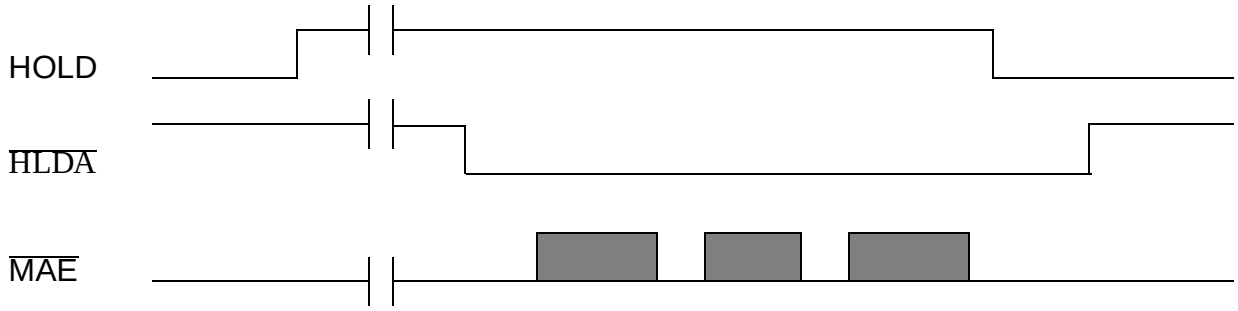


Figure 5-3 MAE Example

After receiving the active $\overline{HDLA}$, the External Master should start driving the Command signals ($\overline{IOR}$, $\overline{IOW}$, $\overline{MEMR}$ and $\overline{MEMW}$), the data (for write cycles) and the address. The address will be enabled/disabled by $\overline{MAE}$ as appropriate.

The SA[31:0] signals in Figure 5-4, "External Master DRAM Access," are the signals connected to the NS486SXL pins SA[31:0]. The timing diagram shows the release of SA[31:0] to the External Master when $\overline{HLDA}$ goes active low. Then, for DRAM accesses, some time after the beginning of each access, the NS486SXL will de-assert $\overline{MAE}$ high and will take over control of SA[31:0] again. This allows the NS486SXL's DRAM controller the ability to multiplex the DRAM row and column addresses out onto SA[12:1] (Note: SA[31:13] and SA[0] will also be driven out of the NS486SXL at this time, but will serve no purpose other than to prevent these signals from floating.)

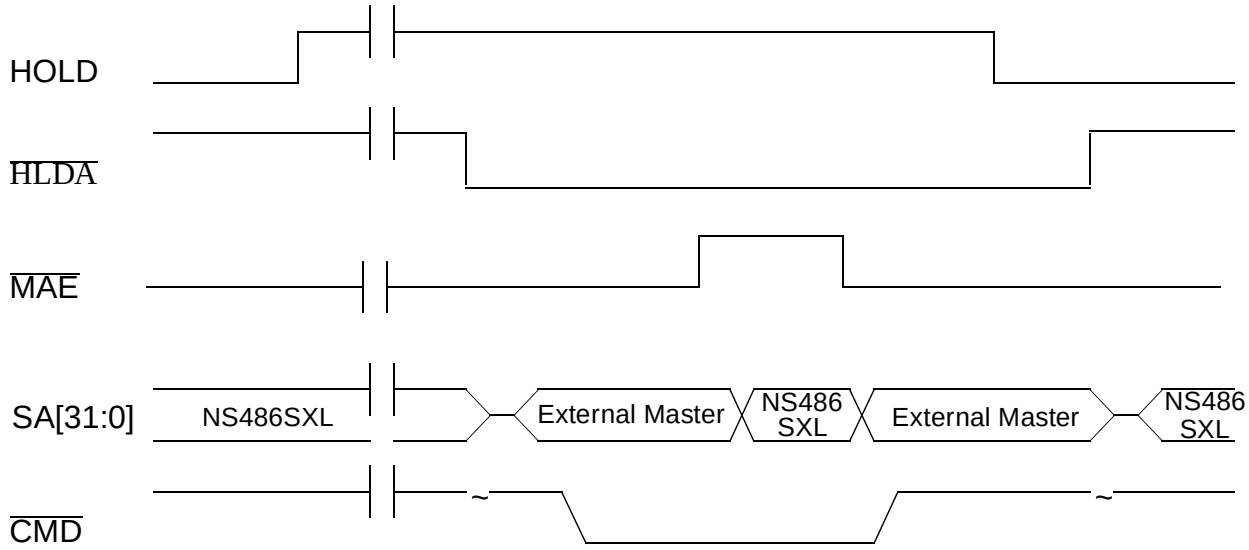


Figure 5-4 External Master DRAM Access

After the DRAM access is completed, the NS486SXL TRI-STATEs the SA[31:0] and reassert $\overline{\text{MAE}}$ active low, allowing the External Master to drive SA[31:0] again in preparation for the next cycle.

For non-DRAM accesses, the NS486SXL does not de-assert $\overline{\text{MAE}}$ during a cycle and the External Master is responsible for maintaining a constant address. Figure 5-5, "Non-DRAM External Master Access," shows a non-DRAM access and the relationship of $\overline{\text{MAE}}$ and SA[31:0].

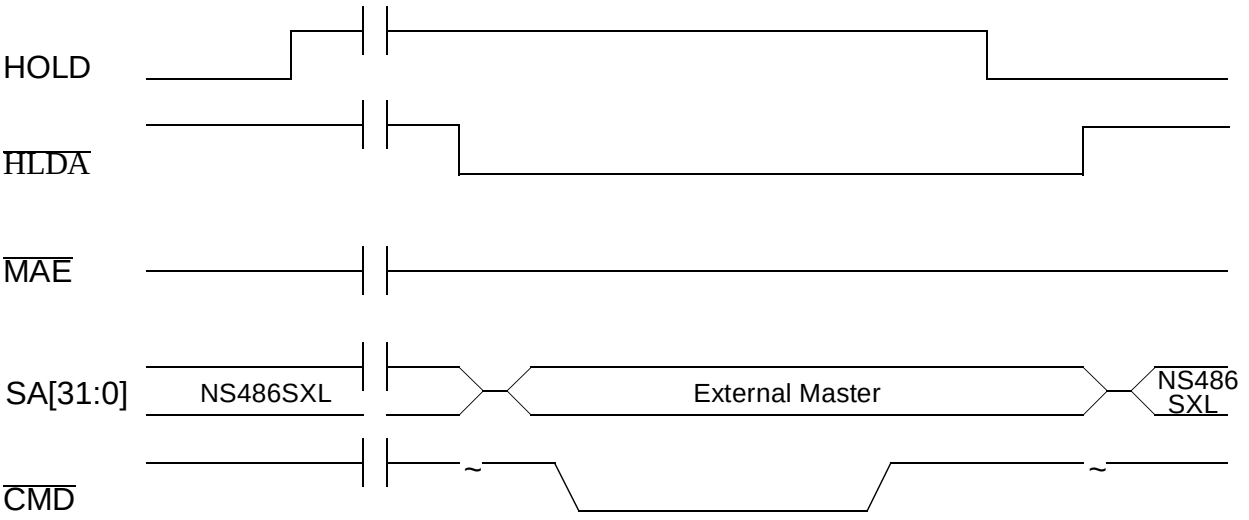


Figure 5-5 Non-DRAM External Master Access

The '~'s on the $\overline{\text{CMD}}$ waveform, '~' indicates the release of the command strobes, which should be pulled inactive high by resistors. It is the responsibility of the NS486SXL and the External Master to only release control of the command strobes after the present driving device has driven them high. In other words, the External Master should not rely on the pullup resisters to pull the command strobes from an active low state to their inactive high state. The present driver of these signals must drive the signals high; then and only then may the present driver release the signal and allow it to be maintained inactive high via the pullup resistor.

For External Master write cycles, the maximum delay from $\overline{CMD}$ being asserted low until valid data is stable on SD[15:0] will be specified. As a preliminary specification, this time will be within one NS486SXL CPU clock period (that would be 40 nsec when operating at 25 MHz).

For write data hold time, it is guaranteed that if data meets a 0 (zero) nsec hold time to $\overline{IOW}$ or $\overline{MEMW}$, both internal peripherals and the DRAM will be written appropriately. The required write data hold time for any other devices on the NS486SXL's external ISA-like bus must be addressed by the system designer.

Figure 5-6, "External Master DRAM Write," shows an External Master write access to DRAM.

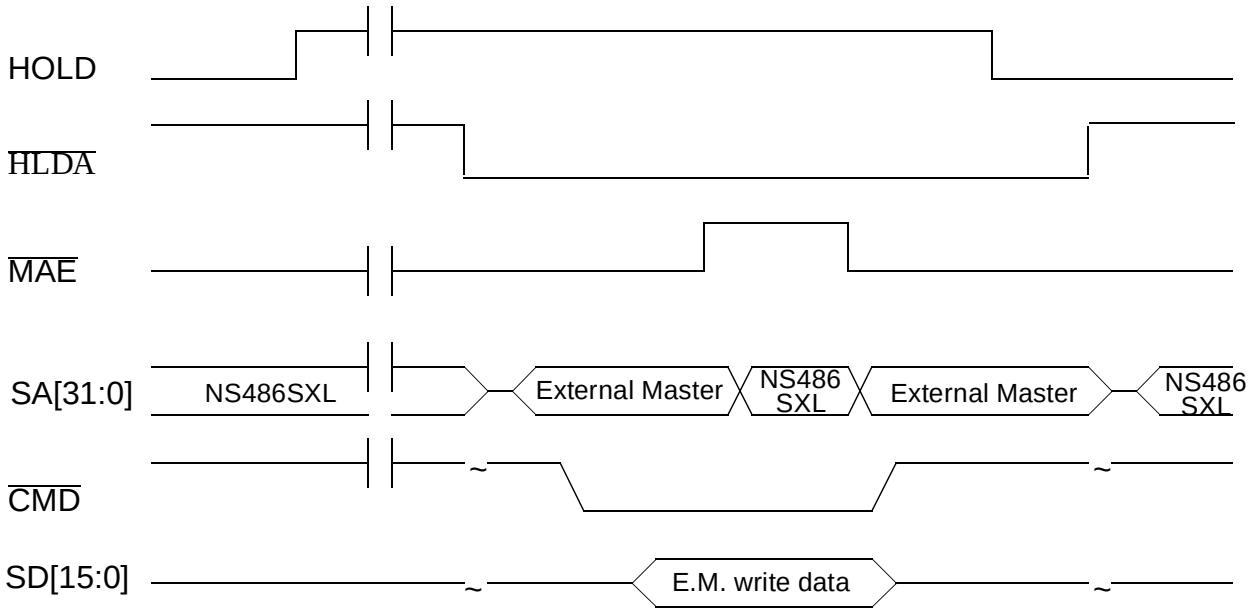


Figure 5-6 External Master DRAM Write

Figure 5-7, “External Master Non-DRAM Write,” shows an External Master write access to non-DRAM.

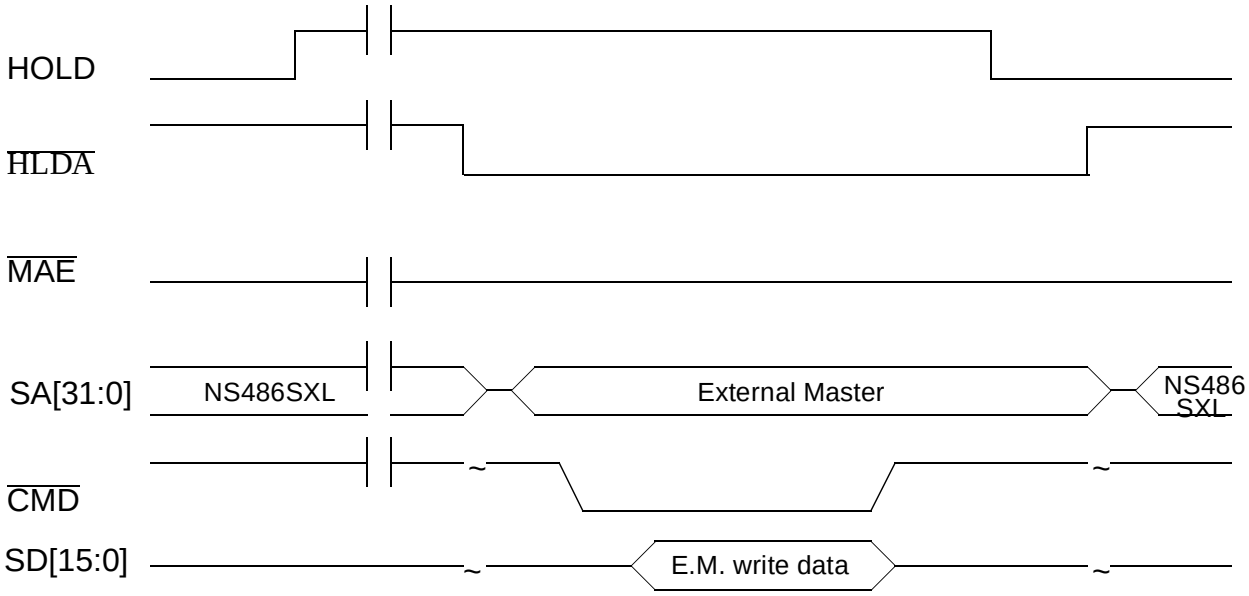


Figure 5-7 External Master Non-DRAM Write

When the External Master gains control of the NS486SXL’s ISA-like bus (i.e. when HLDA goes active low), the open-drain Ready signal (RDY) from the NS486SXL should be able to drive the External Master’s Ready(WAIT) signal active low. For many External Masters, their Ready signal will be a shared open-collector signal. It is the responsibility of the system designer to properly interface the NS486SXL’s RDY to the External Master’s Ready feedback signal.

To ensure proper operation, the NS486SXL will de-assert its RDY (thus inserting wait states back to the External Master) off of the leading (falling) edge of the External Master generated command strobe.

Next, the NS486SXL will then determine if the External Master’s access is to an internal peripheral, DRAM or an external peripheral on the external ISA-like bus. For all accesses, the NS486SXL is guaranteed to de-asserts RDY (via an open-drain style driver) for as long as necessary to emulate an access generated by the NS486SXL’s CPU. If an external peripheral on the external ISA-like bus wishes to insert additional wait states it may do so by also de-asserting RDY low via its own open-drain (or open-collector) driver.

The External Master must maintain an active command strobe while RDY is de-asserted.

Figure 5-8, “RDY During DRAM Write Access,” shows the de-assertion and assertion of RDY for a DRAM write cycle.

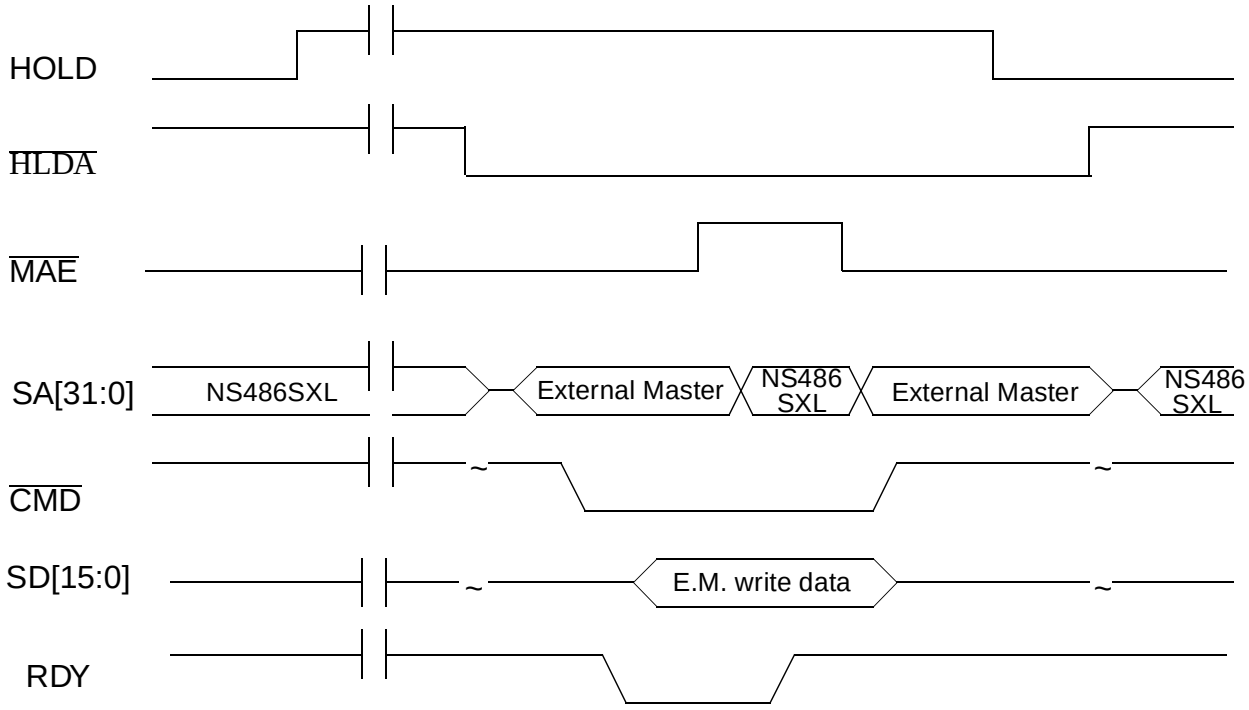


Figure 5-8 RDY During DRAM Write Access

Figure 5-9, “RDY During non-DRAM Write Access,” shows the de-assertion and assertion of RDY for a non-DRAM write cycle.

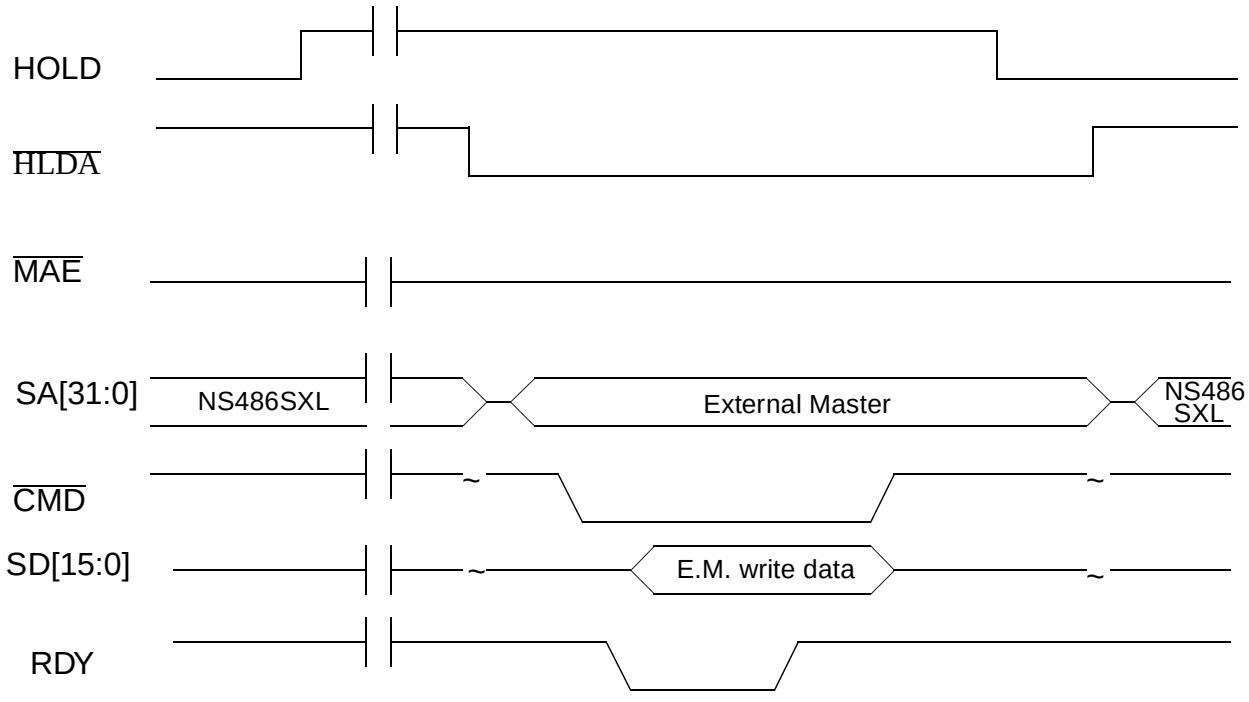


Figure 5-9 RDY During non-DRAM Write Access

For External Master read cycles to internal devices and DRAM, it is guaranteed that the NS486SXL will drive valid data onto SD[15:0] before RDY is asserted high.

For External Master read cycles to external ISA-like bus peripherals, it is the responsibility of the system designer to make sure that proper data setup times are met.

Figure 5-10, “RDY During DRAM Read,” shows the use of RDY for a DRAM read cycle.

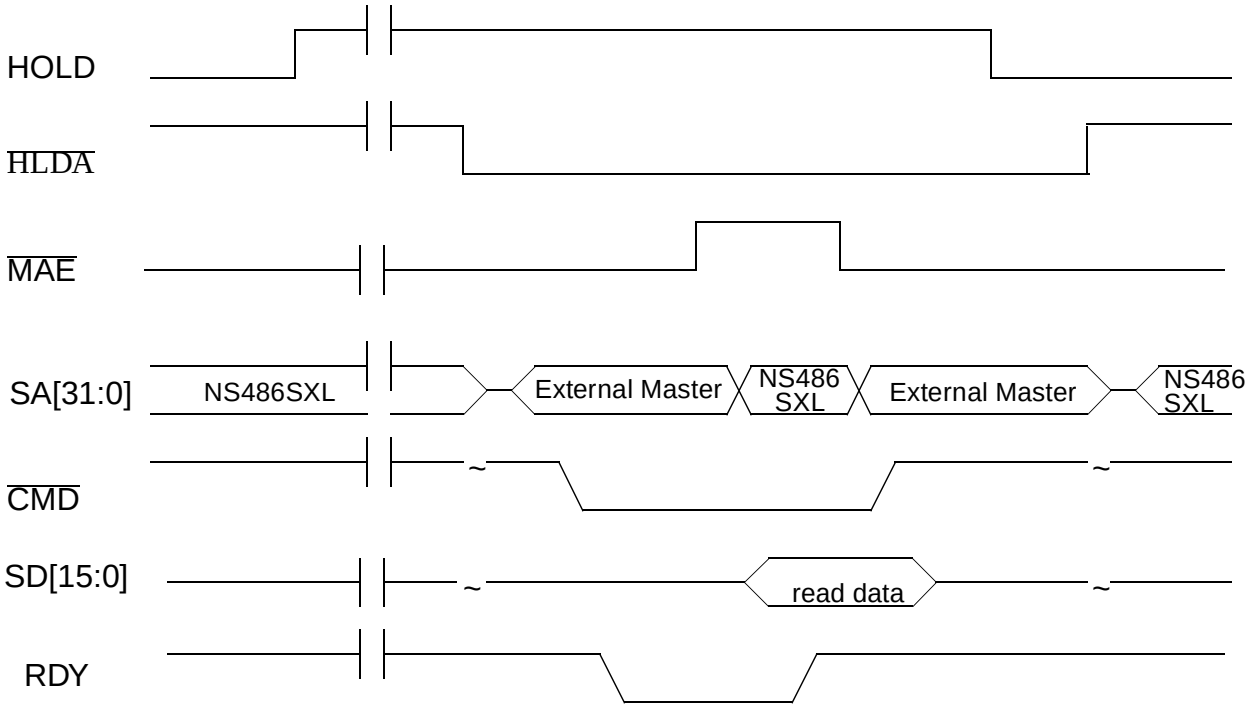


Figure 5-10 RDY During DRAM Read

Figure 5-11, "RDY During non-DRAM Read," shows the de-assertion and assertion of RDY for a non-DRAM read cycle.

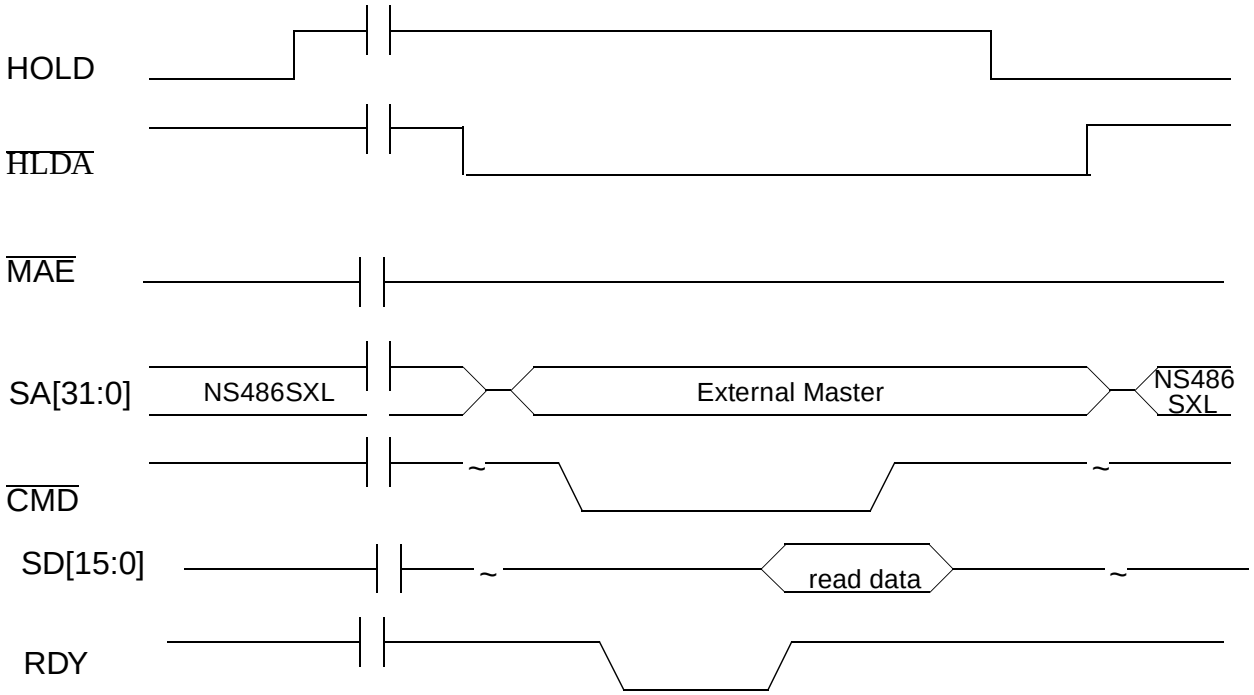


Figure 5-11 RDY During non-DRAM Read

5.2.1 Notes on HOLD Request

It is the responsibility of the External Master to guarantee the proper address hold time for all accesses including the last access. This means that the External Master is not allowed to de-assert its Hold Request (HOLD) until the appropriate system address hold time is met.

If the HOLD is removed before the last command completes, then the HLDA signal will remain asserted (low) until the particular CMD signal (I/O or Memory, read or write) is removed.

Note: The external master is not allowed to re-assert HOLD until it recognizes HLDA inactive.

5.2.2 Notes on RDY

The RDY pin on the NS486SXL is an open-drain output in external master mode (whenever HLDA is active). In external master mode, the System Bus Interface on the 'SXL is not responsible for inserting wait states if an external peripheral holds RDY low. It is the responsibility of the external master to recognize that RDY is low and insert the wait states itself.

5.2.3 Notes on CS16

The CS16 pin on the NS486SXL is an open-drain output in external master mode. In external master mode, the 'SXL will drive CS16 active low any time the DRAM or an internal peripheral access is accessed by the external master.

Note: The CS16 pin is also driven low whenever a programmed Chip Select is set to force a 16-bit access as well. This will occur in both normal mode and external master mode of operation.

5.2.4 Notes on BDIR

BDIR is provided to reduce the glue logic involved in supporting data bus buffers on the 'SXL bus. It has been designed to work in both normal and external master mode. When BDIR is a "1" it indicates that the data on the 'SXL data pins should be driven out onto the buffered ISA-like bus. When BDIR is a "0" the data bus pins should be driven by the buffers from the buffered ISA-like bus. Programmable chip selects have the ability to select whether the peripheral they access is located locally (directly on the pins of the 'SXL) or remotely (on the buffered ISA-like bus); this

feature modifies generation of the BDIR signal as described below.

Whenever $\overline{\text{HLDA}}$ is inactive high the BDIR will be a "1" by default, except during the following case:

- The System Bus Interface is in the Command Delay or the Command States, and a remote, external, non-DRAM read access is taking place.

Whenever $\overline{\text{HLDA}}$ is active low the BDIR will be a "0" by default except during the following case:

- During a read access (IOR or MEMR), the read is from the DRAM, an internal peripheral or a local peripheral, and the 'SXL has not detected the end of the external master command strobe.

5.3 Performance Considerations

The External Master interface is an asynchronous interface. This implies that the performance of the interface is dependent on when the external master provides key signals relative to the internal CPU clock used for synchronization. In the following diagrams, some example accesses are detailed. The actual arbitration for the bus (HOLD request and $\overline{\text{HLDA}}$ assertion) are not shown; the diagrams just illustrate the actual external master access.

There are certain key points in the access where additional CPU clock cycles can result; these are denoted by IDLE, STATE 2 and STATE 4 in the diagrams. Additional IDLEs can result until the 'SXL successfully samples the beginning of an external master command strobe. Additional STATE 2 cycles result until the 'SXL has successfully sampled the internal RDY signal from the DRAM controller or System Bus Interface (from accessing internal peripherals or programmed chip selects). Finally, additional STATE 4 cycles result when the External Master does not remove the command strobe quickly.

There is a minimum command strobe recovery time between each External Master access. This recovery time is shortened by specific logic to watch for the rising edge of the command strobes; the actual required recovery time will be less than one CPU clock period (40 nsec at 25 MHz), and is likely limited by the required address (SA[31:0]) setup to the assertion of the command strobe time requirement. See Figure 5-16, "Fast Back-to-Back Accesses," for an example of quick command strobe toggling.

Figure 5-12, "Shortest Possible System Bus Access," shows the shortest possible access to an internal peripheral (write) or an external ISA-like peripheral (read or write).

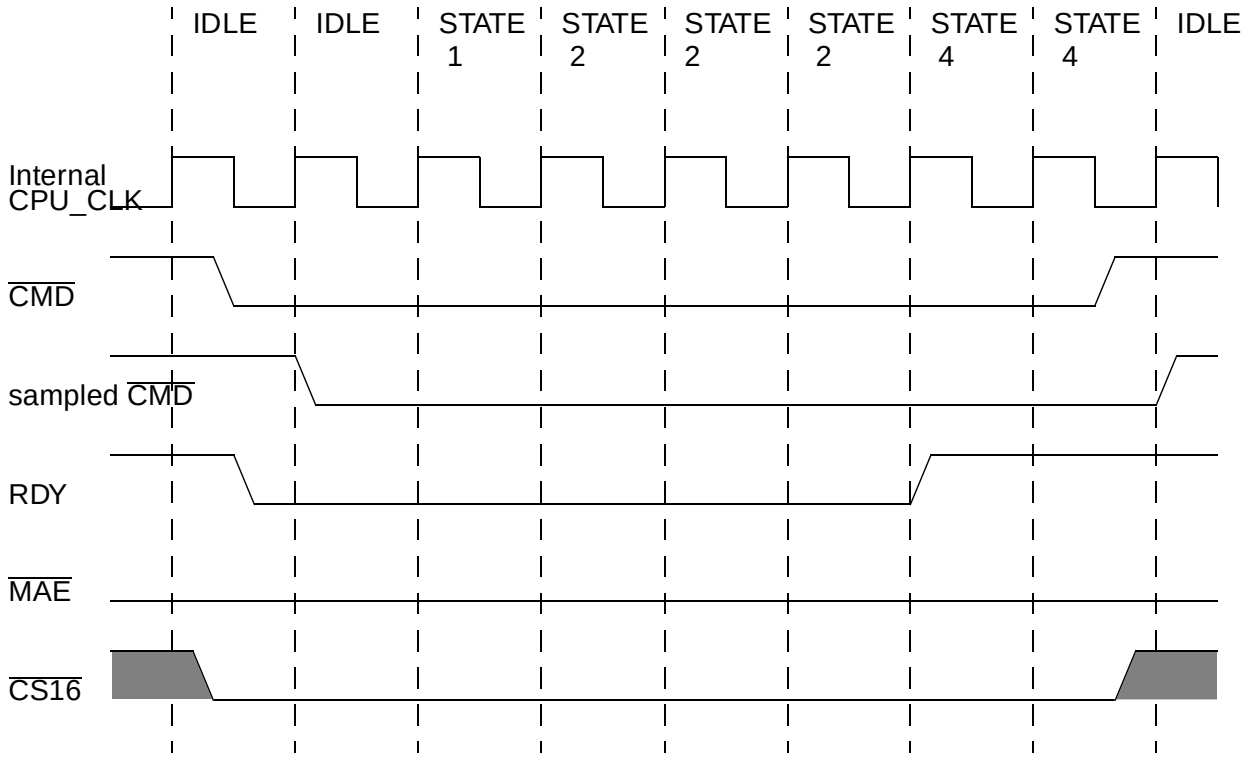


Figure 5-12 Shortest Possible System Bus Access

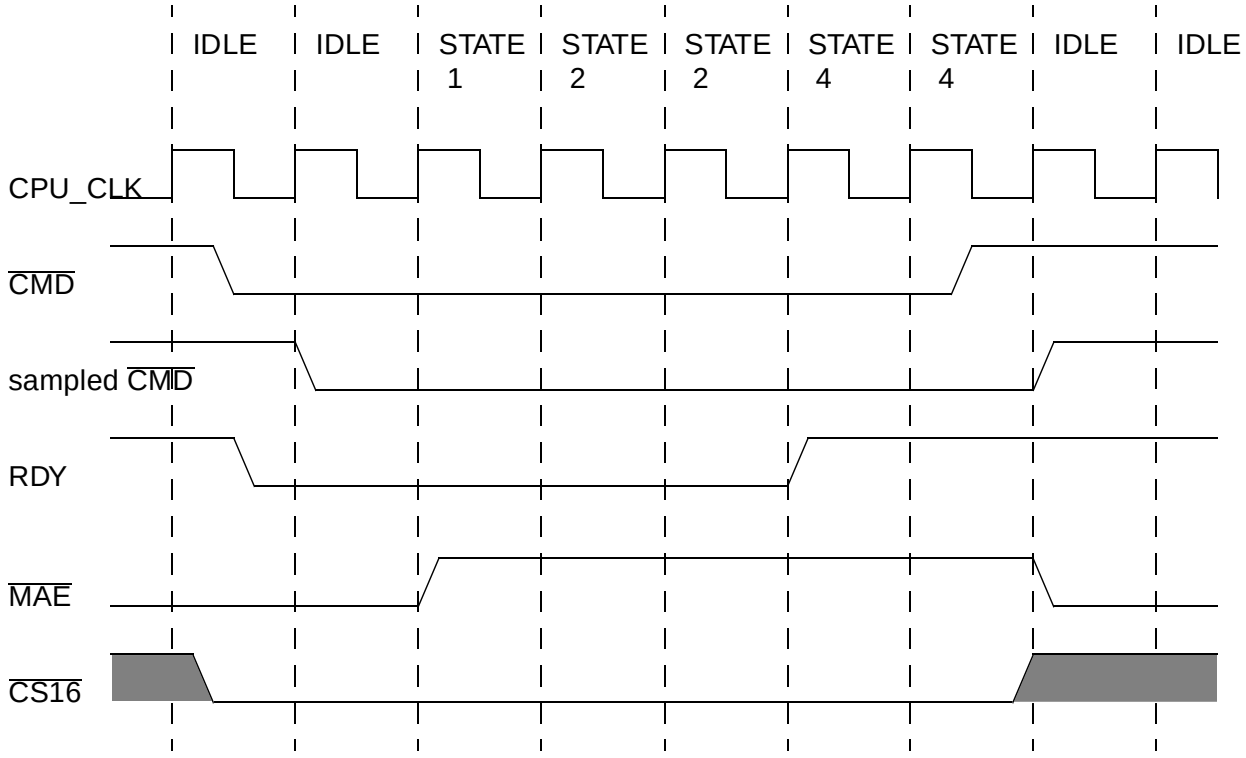


Figure 5-13 DRAM Page Hit Write Access

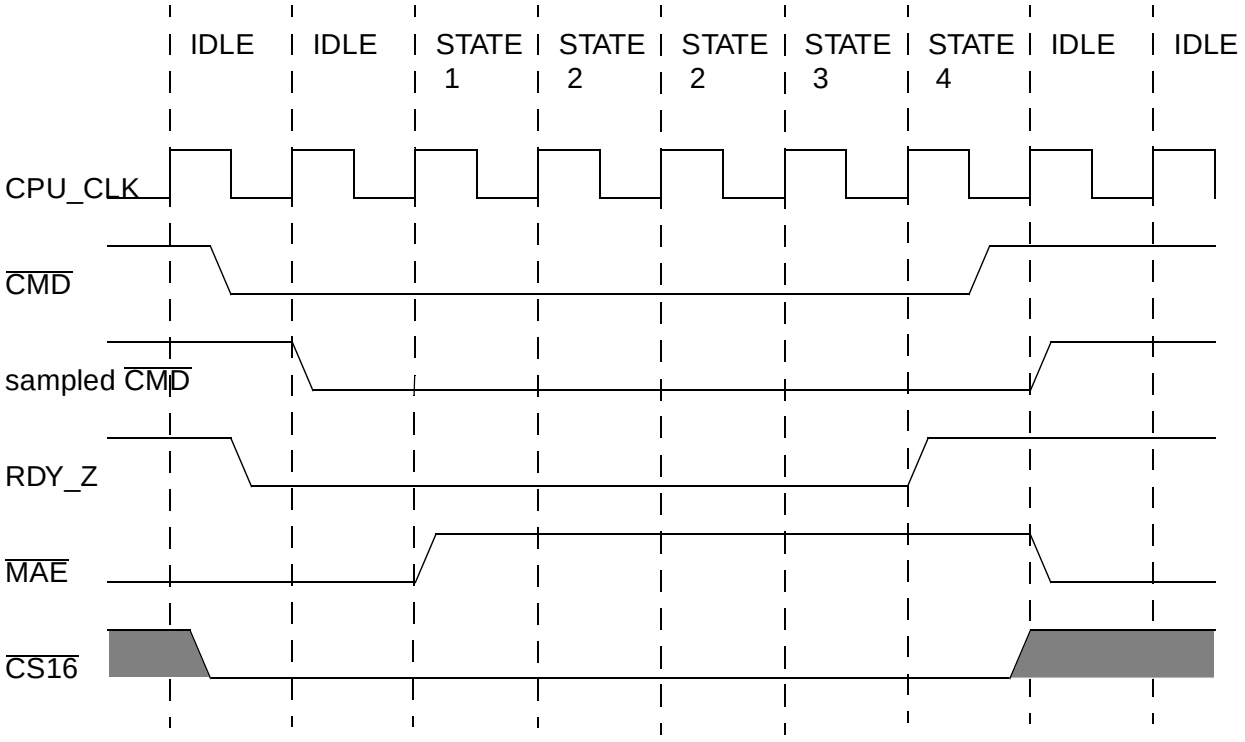


Figure 5-14 DRAM Page Hit Read Access

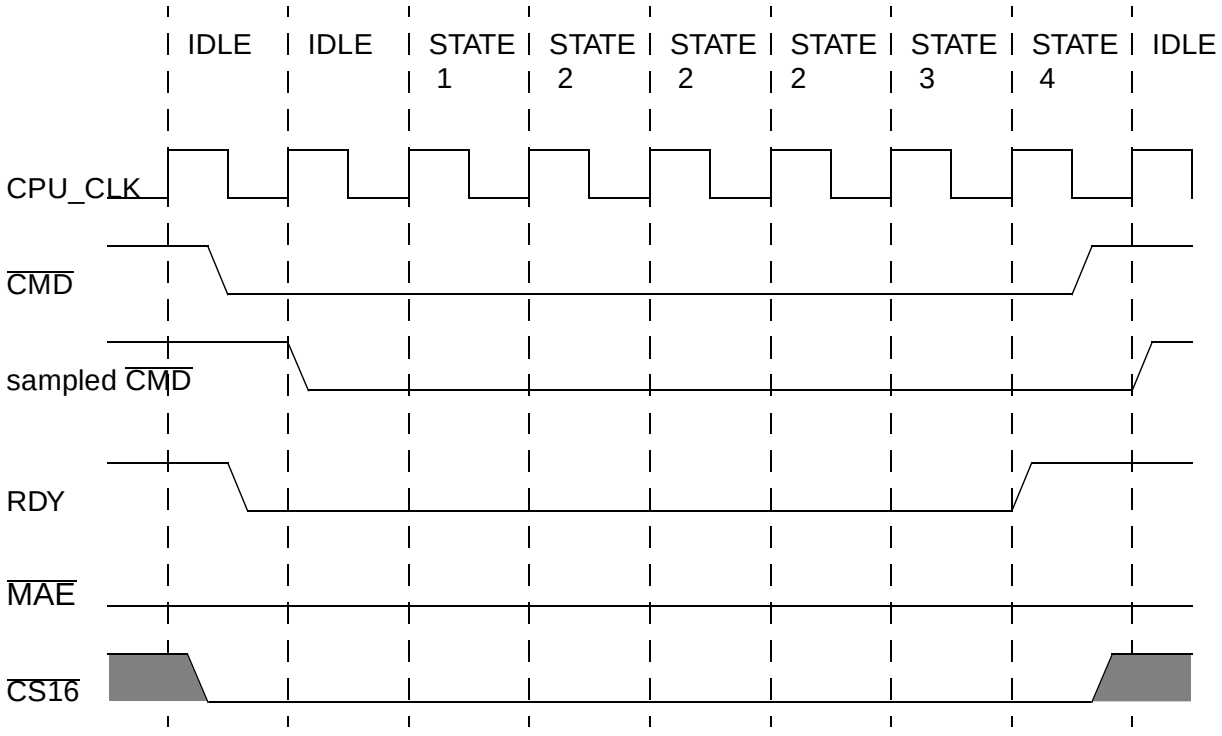


Figure 5-15 Read Access to Internal Peripheral

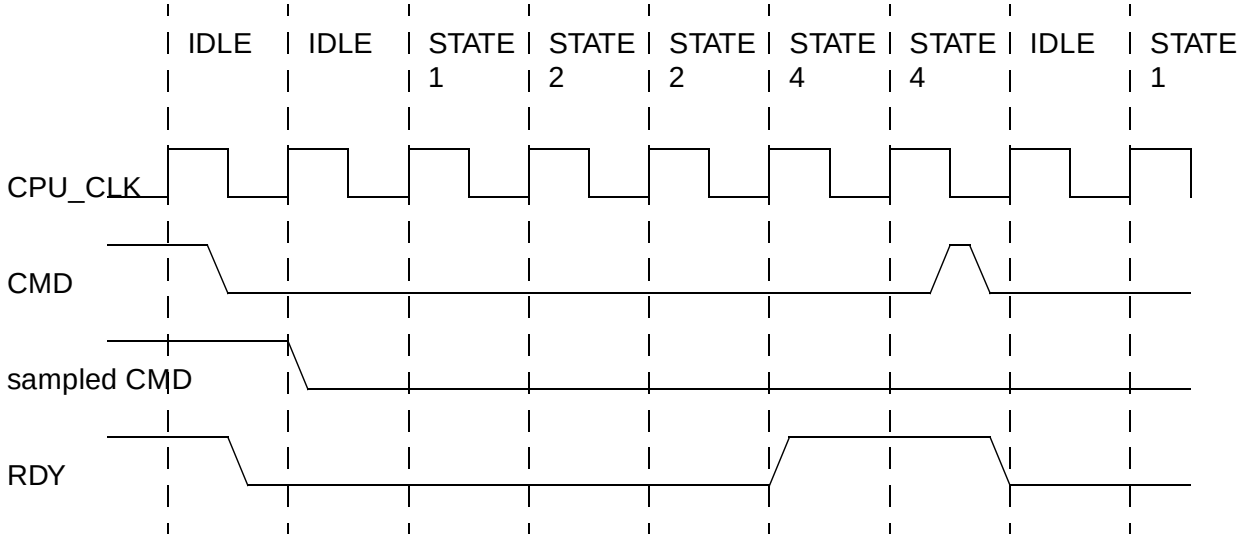


Figure 5-16 Fast Back-to-Back Accesses

This Page Intentionally Left Blank.

6.0 Reconfigurable I/O (RIO)

6.1 Reconfigurable I/O Functional Description

If the **NS486SXL's** UART, MICROWIRE/Access.bus interface, or Auxilary Processor peripheral functions are not being used, the I/O pins associated with them can be reconfigured as general purpose bidirectional I/O pins. In addition, the pins associated with Chip Selects 4-1 (CS4-CS1), the Timer (T0 and T1) or interrupt request pins (IRQ7-0) may also be configured as general purpose bidirection I/O pins. Up to 28 pins, on a pin-by-pin basis can be reconfigured for this purpose. This capability makes the **NS486SXL** extremely versatile and capable of supporting many different end product configurations with a single **NS486SXL** device.

The programming model for the RIO pins has been kept compatible with the NS486SXF wherever possible.

Reconfigurable I/O lines are controlled by two registers: the Reconfigurable I/O Control Register at I/O map address EFC0h, and the 32-bit Data Direction Register at I/O map addresses EFC4h-EFC7h.

6.1.1 Reconfigurable I/O Control Register

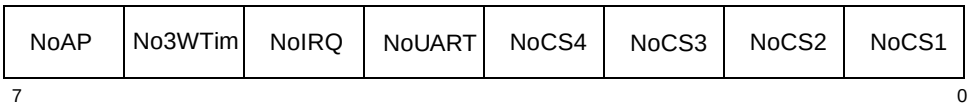


Figure 6-1 Reconfigurable I/O Control Register

This eight bit read/write register is loacated at I/O address EFC0h and its reset value is 00h.

Bit 7: NoAP
Reset value is 0.
0 = Normal Auxilary Processor functionality.
1 = Change the Auxilary Processor pin functionality to correspond to General Purpose I/O bits 31-30.

Bit 6: No3WTim
Reset value is 0.
0 = Normal MICROWIRE/Access.bus functionality.
As well as normal Timer support pin functionality.
1 = Change the MICROWIRE/Access.bus and Timer pin functionality to correspond to General Purpose I/O bits 29-24.

Bit 5: NoIRQ
Reset value is 0.
0 = Normal Interrupt Request functionality.
1 = Allow the Interrupt Request pin functionality to correspond to General Purpose I/O bits 23-16.
NOTE: When this bit is a "1", it does not prevent the corresponding pins (IRQ7 - IRQ0) from operating as interrupt request inputs, as long as their corresponding Data Direction Register bit is a zero. When the corresponding Data Direction Register bit is a one, these pins may still be used as interrupt request sources for the internal Interrupt Controllers, but the source of the signal will be from the corresponding RIO Data Port Out Register bit.

Bit 4: NoUART
Reset value is 0.
0 = Normal UART functionality.
1 = Change the UART pin functionality to correspond to General Purpose I/O 11-4 bits .

Bit 3: NoCS4
Reset value is 0.

0 = Normal Chip Select 4, CS4, functionality.
1 = Change the CS4 pin functionality to correspond to General Purpose I/O bit 3.

Bit 2: NoCS3
Reset value is 0.
0 = Normal Chip Select 3, CS3, functionality.
1 = Change the CS3 pin functionality to correspond to General Purpose I/O bit 2.

Bit 1: NoCS2
Reset value is 0.
0 = Normal Chip Select 2, CS2, functionality.
1 = Change the CS2 pin functionality to correspond to General Purpose I/O bit 1.

Bit 0: NoCS1
Reset value is 0.
0 = Normal Chip Select 1, CS1, functionality.
1 = Change the CS1 pin functionality to correspond to General Purpose I/O bit 0.

6.1.2 Data Direction Register

Each bit in the Data Direction Register determines whether the corresponding pin will be configured as an input or an output. If a bit in the Data Direction Register is a one, the associated pin will be driven as an output by the value in the corresponding value in the Data Port Out register. A zero in a Data Direction register bit makes the corresponding pin an input, and its value can be read by reading the corresponding bit in the Data Port In register.

The 32-bit Data Port In Register is located at address EFC8h-EFCBh in the I/O map. These registers are read-only and always read the value of their associated pins.

The 32-bit Data Port Out Register is located at address EFCCh-EFCFh in the I/O map. These registers are both writable and readable. The value stored in this register is driven onto the corresponding pins if the associated RIO Control bit is set as well as the corresponding RIO Data Direction bit is set to one.

Note that whether a given pin is a general purpose I/O pin or not is determined by the value of the corresponding bit in the Reconfigurable I/O Control Register.

During a system reset the bits in the Data Direction register are set to zeros. This 32-bit register is located at IO addresses EFC4h - EFC7h. The Data Direction register bits 7-0 are located at I/O address EFC4h. The Data Direction register bits 15-8 are located at I/O address EFC5h. The Data Direction register bits 23-16 are located at I/O address EFC6h. The Data Direction register bits 31-24 are located at I/O address EFC7h.

The following table shows the RIO pins and their associated register/bits.

Table 6-2: RIO Pins and Associated Register Bits

I/O Function	Pin #	Name	RIO Ctl. Reg. Bit	Data Dir. Reg. Bit
Aux. Proc.	54	$\overline{\text{DRV}}$	7	31
Aux. Proc	53	$\overline{\text{EACK}}$	7	30
Aux. Proc.	52	$\overline{\text{EREQ}}$	7	29
MICROWIR/ Access.bus	20	SCLK	6	28
MICROWIR/ Access.bus	19	SI	6	27
MICROWIR/ Access.bus	18	SO	6	26
Timer	32	T1	6	25
Timer	31	T0	6	24
Interrupt Cntl.	46	IRQ7	5	22
Interrupt Cntl.	47	IRQ6	5	22
Interrupt Cntl.	57	IRQ5	5	21
Interrupt Cntl.	58	IRQ4	5	20
Interrupt Cntl.	59	IRQ3	5	19
Interrupt Cntl.	60	IRQ2	5	18
Interrupt Cntl.	61	IRQ1	5	17
Interrupt Cntl.	62	IRQ0	5	16
Reserved				15

I/O Function	Pin #	Name	RIO Ctl. Reg. Bit	Data Dir. Reg. Bit
Reserved				14
Reserved				13
Reserved				12
UART	30	$\overline{DTR}$	4	11
UART	29	$\overline{DSR}$	4	10
UART	28	$\overline{RTS}$	4	9
UART	27	$\overline{RI}$	4	8
UART	26	$\overline{CTS}$	4	7
UART	25	$\overline{DCD}$	4	6
UART	34	RX	4	5
UART	35	UCLK	4	4
CS4	43	$\overline{CS[4]}$	3	3
CS3	44	$\overline{CS[3]}$	2	2
CS2	45	$\overline{CS[2]}$	1	1
CS1	63	$\overline{CS[1]}$	0	0

6.1.2.1 Data Direction Register Definitions

Four eight bit registers provide for upto 32 bits of RIO. On the 'SXL only 29 bits are used for RIO pins. The operation of each bit is defined below.

DD_31	DD_30	DD_29	DD_28	DD_27	DD_26	DD_25	DD_24
7							0

Figure 6-3 Reconfigurable I/O Data Direction Register (bits 31-24)

This 8-bit read/write register resides at I/O Address EFC7h and its reset value is 00h.

Bit 31: DD_31 — Data Direction bit 31. This bit is only functional if bit 7 of the RIO Control register is set to a one (i.e. NoAP = 1).

NoAP	DD_31	Operation
0	X	$\overline{DRV}$ (pin 54) remains a Auxilary Processor output
1	0	$\overline{DRV}$ (pin 54) becomes a GPIO input, which may be read via Data Port In bit 31.
1	1	$\overline{DRV}$ (pin 54) becomes a GPIO output, which is driven by the data in Data Port Out bit 31

Bit 30: DD_30 — Data Direction bit 30. This bit is only functional if bit 7 of the RIO Control register is set to a one (i.e. NoAP = 1).

NoAP	DD_30	Operation
0	X	$\overline{EACK}$ (pin 53) remains a Auxilary Processor output
1	0	$\overline{EACK}$ (pin 53) becomes a GPIO input, which may be read via Data Port In bit 30.
1	1	$\overline{EACK}$ (pin 53) becomes a GPIO output, which is driven by the data in Data Port Out bit 30

Bit 29: DD_29 — Data Direction bit 29. This bit is only functional if bit 7 of the RIO Control register is set to a one (i.e. NoAP = 1).

NoAP	DD_29	Operation
0	X	$\overline{EREQ}$ (pin 52) remains a Auxilary Processor input

NoAP	DD_29	Operation
1	0	$\overline{\text{EREQ}}$ (pin 52) becomes a GPIO input, which may be read via Data Port In bit 29.
1	1	$\overline{\text{EREQ}}$ (pin 52) becomes a GPIO output, which is driven by the data in Data Port Out bit 29.

Bit 28: DD_28 — Data Direction bit 28. This bit is only functional if bit 6 of the RIO Control register is set to a one (i.e. No3WTim = 1).

No3WTim	DD_28	Operation
0	X	SCLK (pin 20) remains a MICROWIRE/ Access.bus output.
1	0	SCLK (pin 20) becomes a GPIO input, which may be read via Data Port In bit 28.
1	1	SCLK (pin 20) becomes a GPIO output, which is driven by the data in Data Port Out bit 28.

Bit 27: DD_27 — Data Direction bit 27. This bit is only functional if bit 6 of the RIO Control register is set to a one (i.e. No3WTim = 1).

No3WTim	DD_27	Operation
0	X	SI (pin 19) MICROWIRE/ Access.bus input
1	0	SI (pin 19) becomes a GPIO input which may be read via Data Port In bit 27.
1	1	BSI (pin 19) becomes a GPIO output, which is driven by the data in Data Port Out bit 27.

Bit 26: DD_26 — Data Direction bit 26. This bit is only functional if bit 6 of the RIO Control register is set to a one (i.e. No3WTim = 1).

No3WTim	DD_26	Operation
0	X	SO (pin 18) MICROWIRE/ Access.bus output
1	0	SO (pin 18) becomes a GPIO input which may be read via Data Port In bit 26.
1	1	SO (pin 18) becomes a GPIO output, which is driven by the data in Data Port Out bit 26.

Bit 25: DD_25 — Data Direction bit 25. This bit is only functional if bit 6 of the RIO Control register is set to a one (i.e. No3WTim = 1).

No3WTim	DD_25	Operation
0	X	T1 (pin 32) remains a Timer I/O
1	0	T1 (pin 32) becomes a GPIO input which may be read via Data Port In bit 25.
1	1	T1 (pin 32) becomes a GPIO output, which is driven by the data in Data Port Out bit 25.

Bit 24: DD_24 — Data Direction bit 24. This bit is only functional if bit 6 of the RIO Control register is set to a one (i.e. No3WTim = 1).

No3WTim	DD_24	Operation
0	X	T0 (pin 31) remains a Timer I/O
1	0	T0 (pin 31) becomes a GPIO input which may be read via Data Port In bit 24.
1	1	T0 (pin 31) becomes a GPIO output, which is driven by the data in Data Port Out bit 24.

.

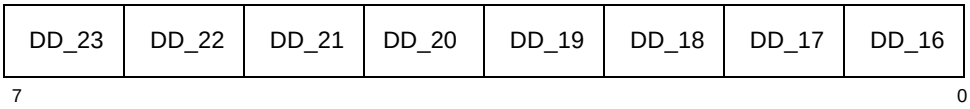


Figure 6-4 Reconfigurable I/O Data Direction Register (bits 23-16)

This 8-bit read/write register is located at I/O address EFC6h and its reset value is 00h.

DD_23 — Data Direction bit 23. This bit is only functional if bit 5 of the RIO Control register is set to a one (i.e. NoIRQ = 1).

NoIRQ	DD_23	Operation
0	X	IRQ7 (pin 46) remains a Interrupt Request input. NOTE: This pin like all GPIO pins may be read via its Data Port In bit at any time, thus this entry and the next entry for this bit are functional equavalents.
1	0	IRQ7 (pin 46) becomes a GPIO input, which may be read via Data Port In bit 23. NOTE: This pin may continue to operate as an Interrupt Request input even when it is configured as a GPIO input.
1	1	IRQ7 (pin 46) becomes a GPIO output, which is driven by the data in Data Port Out bit 23.

Bit 22: DD_22 — Data Direction bit 22. This bit is only functional if bit 5 of the RIO Control register is set to a one (i.e. NoIRQ = 1).

NoIRQ	DD_22	Operation
0	X	IRQ6 (pin 47) remains a Interrupt Request input. NOTE: This pin like all GPIO pins may be read via its Data Port In bit at any time, thus this entry and the next entry for this bit are functional equavalents.
1	0	IRQ6 (pin 47) becomes a GPIO input, which may be read via Data Port In bit 22. NOTE: This pin may continue to operate as an Interrupt Request input even when it is configured as a GPIO input.
1	1	IRQ6 (pin 47) becomes a GPIO output, which is driven by the data in Data Port Out bit 22.

Bit 21: DD_21 — Data Direction bit 21. This bit is only functional if bit 6 of the RIO Control register is set to a one (i.e. NoIRQ = 1).

NoIRQ	DD_21	Operation
0	X	IRQ5 (pin 57) remains a Interrupt Request input. NOTE: This pin like all GPIO pins may be read via its Data Port In bit at any time, thus this entry and the next entry for this bit are functional equavalents.

NoIRQ	DD_21	Operation
1	0	IRQ5 (pin 57) becomes a GPIO input, which may be read via Data Port In bit 21. NOTE: This pin may continue to operate as an Interrupt Request input even when it is configured as a GPIO input.
1	1	IRQ5 (pin 57) becomes a GPIO output, which is driven by the data in Data Port Out bit 21.

Bit 20: DD_20 — Data Direction bit 20. This bit is only functional if bit 5 of the RIO Control register is set to a one (i.e. NoIRQ = 1).

NoIRQ	DD_20	Operation
0	X	IRQ4 (pin 58) remains a Interrupt Request input. NOTE: This pin like all GPIO pins may be read via its Data Port In bit at any time, thus this entry and the next entry for this bit are functional equavalents.
1	0	IRQ4 (pin 58) becomes a GPIO input, which may be read via Data Port In bit 20. NOTE: This pin may continue to operate as an Interrupt Request input even when it is configured as a GPIO input.
1	1	IRQ4 (pin 58) becomes a GPIO output, which is driven by the data in Data Port Out bit 20.

Bit 19: DD_19 — Data Direction bit 19. This bit is only functional if bit 5 of the RIO Control register is set to a one (i.e. NoIRQ = 1).

NoIRQ	DD_19	Operation
0	X	IRQ3 (pin 59) remains a Interrupt Request input. NOTE: This pin like all GPIO pins may be read via its Data Port In bit at any time, thus this entry and the next entry for this bit are functional equavalents.
1	0	IRQ3 (pin 59) becomes a GPIO input, which may be read via Data Port In bit 19. NOTE: This pin may continue to operate as an Interrupt Request input even when it is configured as a GPIO input.
1	1	IRQ3 (pin 59) becomes a GPIO output, which is driven by the data in Data Port Out bit 19.

Bit 18: DD_18 — Data Direction bit 18. This bit is only functional if bit 5 of the RIO Control register is set to a one (i.e. NoIRQ = 1).

NoIRQ	DD_18	Operation
0	X	IRQ2 (pin 60) remains a Interrupt Request input. NOTE: This pin like all GPIO pins may be read via its Data Port In bit at any time, thus this entry and the next entry for this bit are functional equavalents.
1	0	IRQ2 (pin 60) becomes a GPIO input, which may be read via Data Port In bit 18. NOTE: This pin may continue to operate as an Interrupt Request input even when it is configured as a GPIO input.

NoIRQ	DD_18	Operation
1	1	IRQ2 (pin 60) becomes a GPIO output, which is driven by the data in Data Port Out bit 18.

Bit 16: DD_16 — Data Direction bit 16. This bit is only functional if bit 5 of the RIO Control register is set to a one (i.e. NoIRQ = 1).

Bit 17: DD_17 — Data Direction bit 17. This bit is only functional if bit 5 of the RIO Control register is set to a one (i.e. NoIRQ = 1).

NoIRQ	DD_17	Operation
0	X	IRQ1 (pin 61) remains a Interrupt Request input. NOTE: This pin like all GPIO pins may be read via its Data Port In bit at any time, thus this entry and the next entry for this bit are functional equavalents.
1	0	IRQ1 (pin 61) becomes a GPIO input, which may be read via Data Port In bit 17. NOTE: This pin may continue to operate as an Interrupt Request input even when it is configured as a GPIO input.
1	1	IRQ1 (pin 61) becomes a GPIO output, which is driven by the data in Data Port Out bit 17.

NoIRQ	DD_16	Operation
0	X	IRQ0 (pin 62) remains a Interrupt Request input. NOTE: This pin like all GPIO pins may be read via its Data Port In bit at any time, thus this entry and the next entry for this bit are functional equavalents.
1	0	IRQ0 (pin 62) becomes a GPIO input, which may be read via Data Port In bit 16. NOTE: This pin may continue to operate as an Interrupt Request input even when it is configured as a GPIO input.
1	1	IRQ0 (pin 62) becomes a GPIO output, which is driven by the data in Data Port Out bit 16.

.

—	—	—	—	DD_11	DD_10	DD_9	DD_8
7							0

Figure 6-5 Reconfigurable I/O Data Direction Register (bits 15-8)

This 8-bit read/write register is located at I/O address EFC5h and its reset value is 00h. The upper four bits of this register will always be read as zeros; writing to the upper four bits of this register will have no effects.

Bit 11: DD_11 — Data Direction bit 11. This bit is only functional if bit 4 of the RIO Control register is set to a one (i.e. NoUART = 1).

Bits 15-12: DD_15-DD_12: Reserved.

NoUART	DD_11	Operation
0	X	$\overline{DTR}$ (pin 30) remains an UART .
1	0	$\overline{DTR}$ (pin 30) becomes a GPIO input, which may be read via Data Port In bit 11.
1	1	$\overline{DTR}$ (pin 30) becomes a GPIO output, which is driven by the data in Data Port Out bit 11.

Bit 10: DD_10 — Data Direction bit 10. This bit is only functional if bit 4 of the RIO Control register is set to a one (i.e. NoUART = 1).

NoUART	DD_10	Operation
0	X	$\overline{DSR}$ (pin 29) remains an UART .
1	0	$\overline{DSR}$ (pin 29) becomes a GPIO input, which may be read via Data Port In bit 10.
1	1	$\overline{DSR}$ (pin 29) becomes a GPIO output, which is driven by the data in Data Port Out bit 10.

Bit 9: DD_9 — Data Direction bit 9. This bit is only functional if bit 4 of the RIO Control register is set to a one (i.e. NoUART = 1).

NoUART	DD_9	Operation
0	X	$\overline{RTS}$ (pin 28) remains an UART .
1	0	$\overline{RTS}$ (pin 28) becomes a GPIO input, which may be read via Data Port In bit 9.
1	1	$\overline{RTS}$ (pin 28) becomes a GPIO output, which is driven by the data in Data Port Out bit 9.

Bit 8: DD_8 — Data Direction bit 8. This bit is only functional if bit 4 of the RIO Control register is set to a one (i.e. NoUART = 1).

NoUART	DD_8	Operation
0	X	$\overline{RI}$ (pin 27) remains an UART .
1	0	$\overline{RI}$ (pin 27) becomes a GPIO input, which may be read via Data Port In bit 8.
1	1	$\overline{RI}$ (pin 27) becomes a GPIO output, which is driven by the data in Data Port Out bit 8.

DD_7	DD_6	DD_5	DD_4	DD_3	DD_2	DD_1	DD_0
7							0

Figure 6-6 Reconfigurable I/O Data Direction Register (bits 7-0)

This 8-bit read/write register is located at I/O address EFC4h and its reset value is 00h.

Bit 7: DD_7 — Data Direction bit 7. This bit is only functional if bit 4 of the RIO Control register is set to a one (i.e. NoUART = 1).

NoUART	DD_7	Operation
0	X	$\overline{CTS}$ (pin 26) remains an UART .
1	0	$\overline{CTS}$ (pin 26) becomes a GPIO input, which may be read via Data Port In bit 7.

NoUART	DD_7	Operation
1	1	$\overline{CTS}$ (pin 26) becomes a GPIO output, which is driven by the data in Data Port Out bit 7.

Bit 8: DD_6 — Data Direction bit 6. This bit is only functional if bit 4 of the RIO Control register is set to a one (i.e. NoUART = 1).

NoUART	DD_6	Operation
0	X	$\overline{DCD}$ (pin 25) remains an UART .
1	0	$\overline{DCD}$ (pin 25) becomes a GPIO input, which may be read via Data Port In bit 6.
1	1	$\overline{DCD}$ (pin 25) becomes a GPIO output, which is driven by the data in Data Port Out bit 6.

Bit 5: DD_5 — **D**ata **D**irection bit **5**. This bit is only functional if bit 4 of the RIO Control register is set to a one (i.e. NoUART = 1).

NoUART	DD_5	Operation
0	X	RX (pin 34) remains an UART input.
1	0	RX (pin 34) becomes a GPIO input, which may be read via Data Port In bit 5.
1	1	RX (pin 34) becomes a GPIO output, which is driven by the data in Data Port Out bit 5.

Bit 4: DD_4 — Data Direction bit 4. This bit is only functional if bit 4 of the RIO Control register is set to a one (i.e. NoUART = 1).

NoUART	DD_4	Operation
0	X	UCLK (pin 35) remains an UART output.
1	0	UCLK (pin 35) becomes a GPIO input, which may be read via Data Port In bit 4.
1	1	UCLK (pin 35) becomes a GPIO output, which is driven by the data in Data Port Out bit 4.

Bit 3: DD_3 — Data Direction bit 3. This bit is only functional if bit 3 of the RIO Control register is set to a one (i.e. NoCS4 = 1).

NoCS8	DD_3	Operation
0	X	$\overline{CS}$[4] (pin 43) remains chip select 4 output pin.
1	0	$\overline{CS}$[4] (pin 43) becomes a GPIO input, which may be read via Data Port In bit 3.
1	1	$\overline{CS}$[4] (pin 43) becomes a GPIO output, which is driven by the data in Data Port Out bit 3.

Bit 2: DD_2 — Data Direction bit 2. This bit is only functional if bit 3 of the RIO Control register is set to a one (i.e. NoCS3 = 1).

NoCS7	DD_2	Operation
0	X	$\overline{CS}$[3] (pin 44) remains chip select 3 output pin.
1	0	$\overline{CS}$[3] (pin 44) becomes a GPIO input, which may be read via Data Port In bit 2.
1	1	$\overline{CS}$[3] (pin 44) becomes a GPIO output, which is driven by the data in Data Port Out bit 2.

Bit 1: DD_1 — Data Direction bit 1. This bit is only functional if bit 1 of the RIO Con-

trol register is set to a one (i.e. NoCS2 = 1).

NoCS6	DD_1	Operation
0	X	$\overline{CS}$[2] (pin 45) remains chip select 2 output pin.
1	0	$\overline{CS}$[62 (pin 45) becomes a GPIO input, which may be read via Data Port In bit 1.
1	1	$\overline{CS}$[2] (pin 45) becomes a GPIO output, which is driven by the data in Data Port Out bit 1.

Bit 0: DD_0 — Data Direction bit 0. This bit is only functional if bit 0 of the RIO Control register is set to a one (i.e. NoCS1 = 1).

NoCS5	DD_0	Operation
0	X	$\overline{CS}$[1] (pin 63) remains chip select 1 output pin.
1	0	$\overline{CS}$[1] (pin 63) becomes a GPIO input, which may be read via Data Port In bit 0.
1	1	$\overline{CS}$[1] (pin 63) becomes a GPIO output, which is driven by the data in Data Port Out bit 0.

6.1.2.2 Data Port Out Register

This 32-bit read/write register is located at IO addresses EFCCh - EFCFh. This register provides the data output storage for the General Purpose I/O discussed in the previous Data Direction Register section. When a RIO pin is defined as an output (by disabling the initial I/O function in the RIO Control Register and by setting the associated bit with the value of one in the Data Direction Register), its output value is contained in this 32-bit register.

Bits 7-0 of this register are at I/O address EFCCh.
Bits 15-8 of this register are at I/O address EFCDh. Please note that bits 15-8 will always read as zeros and writing to these four bits will have no effects.
Bits 23-16 of this register are at I/O address EFCEh.
Bits 31-24 of this register are at I/O address EFCFh.

6.1.2.3 Data Port In Register

This 32-bit read only register is located at IO addresses EFC8h - EFCBh. When this register is read, it will return the values currently on the corresponding **NS486SXL** pins described in the prior Data Direction Register section, regardless whether these pins are configured as input only, outputs or under the control of their primary function. Writing to these registers will have no effects.

Bits 7-0 of this register are at I/O address EFC8h.
Bits 15-8 of this register are at I/O address EFC9h. Please note that bits 15-12 will always read as zeros.
Bits 23-16 of this register are at I/O address EFCAh.
Bits 31-24 of this register are at I/O address EFCBh.

7.0 Device Specifications

7.1 DC Electrical Specifications 5V ± 5%

7.1.1 Recommended Operating Conditions

Characteristic	Condition	Symbol	Min	Typ	Max	Unit
Supply Voltage		V_{DD}	4.75	5.0	5.25	V
Operating Temperature		T_A	0		+70	°C
ESD Tolerance	C_{ZAP} = 100 pF R_{ZAP} = 1.5 kΩ (Note 1)		2000			V

7.1.2 Absolute Maximum Ratings (Notes 2 and 3)

Characteristic	Condition	Symbol	Min	Max	Unit
Supply Voltage		V_{DD}, V_{DDA}	-0.5	7.0	V
Input Voltage		V_I	-0.5	V_{DD} + 0.5	V
Output Voltage		V_O	-0.5	V_{DD} + 0.5	V
Storage Temperature		T_{STG}	-65	+165	° C
Lead Temperature Soldering (10 sec.)		T_L		+260	° C

7.1.3 Capacitance: T_A = 25°C, f = 1 MHz

Characteristics	Symbol	Min	Typ	Max	Unit
Input Pin Capacitance	C_{IN}		5	7	pF
Clock Input Capacitance	C_{IN1}		8	10	pF
I/O Pin Capacitance	C_{IO}		10	12	pF
Output Pin Capacitance	C_O		6	8	pF

7.1.4 DC Characteristics (Under Recommended Operating Conditions)

Parameter	Symbol	Condition	Min	Typ	Max	Unit
Input High Voltage	V_{IH}		2.0		V_{DD}	V
Input Low Voltage	V_{IL}		-0.5		0.8	V
V_{DD} Average Supply Current	I_{CC}	V_{IL} = 0.5V V_{IH} = 2.4V No Load		250	350	mA

Note 1: Value based on test complying with NSC SOP5-028 human body model ESD testing using the ETS-910 tester.

Note 2: Absolute Maximum Ratings are those values beyond which damage to the device may occur.

Note 3: Unless otherwise specified all voltages are referenced to ground.

7.1.4.1 External Bus

Parameter	Symbol	Condition	Min	Max	Unit	Notes
Output High Voltage	V_{OH}	I_{OH} = -6mA(Nch Quiet-drive) or I_{OH} = -24mA(High-drive) on: SA12-1, DP1-0, SD15-0 I_{OH} = -12mA on: SA0, SA25-13 [SA0 - min. 10Kohm pullup]	2.4		V	Max load on SA12-1 is 50pF, and SD0-15 is 50pF
Output Low Voltage	V_{OL}	I_{OL} = 20 mA on: SA12-1, DP1-0, SD15-0 I_{OL} = 12mA on: SA0, SA25-13, $\overline{BHE}$		0.4	V	

7.1.4.2 DRAM Control Unit

Parameter	Symbol	Condition	Min	Max	Unit	Notes
Output High Voltage	V_{OH}	I_{OH} = -6mA(Nch Quiet-drive) or I_{OH} = -24mA(High-drive) on: $\overline{RAS}$0-1, $\overline{CASH}$0-1, $\overline{CASL}$0-1, $\overline{WE}$	2.4		V	Max load on $\overline{RAS}$1-0, $\overline{CASH}$1-0, & $\overline{CASL}$1-0 is 63pF.
Output Low Voltage	V_{OL}	I_{OL} = 20mA on: $\overline{RAS}$1-0, $\overline{CASH}$1-0, $\overline{CASL}$1-0, $\overline{WE}$		0.4	V	Max load on $\overline{WE}$ is 50pF.

7.1.4.3 Auxiliary Processor Interface

Parameter	Symbol	Condition	Min	Max	Unit	Notes
Output High Voltage	V_{OH}	I_{OH} = -6mA on: $\overline{EACK}$ I_{OH} = -4mA on: $\overline{DRV}$, $\overline{EREQ}$	2.4		V	
Output Low Voltage	V_{OL}	I_{OL} = 6mA on: $\overline{EACK}$ I_{OL} = 4mA on: $\overline{DRV}$, $\overline{EREQ}$		0.4	V	

7.1.4.4 IrDA Infra Red/UART

Parameter	Symbol	Condition	Min	Max	Unit	Notes
Output High Voltage	V_{OH}	I_{OH} = -100uA I_{OH} = -6mA on: Tx, UCLK, Rx	Vcc - 0.2 2.4		V V	
Output Low Voltage	V_{OL}	I_{OL}= 100uA I_{OL} = 6mA on: Tx, UCLK, Rx		0.2 0.4	V V	

7.1.4.5 External Bus Control

Parameter	Symbol	Condition	Min	Max	Unit	Notes
Output High Voltage	V_{OH}	I_{OH} = -12mA on: $\overline{IOR}$, $\overline{IOW}$, $\overline{MEMR}$, $\overline{MEMW}$ RESET, $\overline{RESET}$, $\overline{CS16}$, $\overline{BHE}$ [$\overline{CS16}$ - min. 10Kohm pullup]	2.4		V	
Output Low Voltage	V_{OL}	I_{OL} = 12mA on: $\overline{IOR}$, $\overline{IOW}$, $\overline{MEMR}$, $\overline{MEMW}$ RESET, $\overline{RESET}$, $\overline{CS16}$, $\overline{BHE}$		0.4	V	

7.1.4.6 Oscillator (CPUX1/CLK)

Parameter	Symbol	Condition	Min	Max	Unit	Notes
Output High Voltage	V_{OH}	I_{OH} = -12mA on: SYSCLK	2.4		V	
Output Low Voltage	V_{OL}	I_{OL} = 12mA on: SYSCLK		0.4	V	
OSCX1 Input High Voltage	V_{IH}		2.0			OSCX2 is the output
OSCX2 Input Low Voltage	V_{IL}			0.4	V	

7.1.4.7 Real Time Clock (RTCX1/CLK)

Parameter	Symbol	Condition	Min	Max	Unit	Notes
RTCX1 Input High Voltage	V_{IH}		2.0			RTCX2 is the output
RTCX1 Input Low Voltage	V_{IL}			0.4	V	
Battery Voltage	V_{BAT}		2.4		V	Lithium Battery
Battery Current	I_{BAT}	V_{BAT} = 3.0 V				

7.1.4.8 Timer

Parameter	Symbol	Condition	Min	Max	Unit	Notes
Output High Voltage	V_{OH}	I_{OH} = -6mA on: T0, T1	2.4		V	
Output Low Voltage	V_{OL}	I_{OL} = 6mA on: T0, T1		0.4	V	

7.1.4.9 General Purpose Chip Selects

Parameter	Symbol	Condition	Min	Max	Unit	Notes
Output High Voltage	V_{OH}	I_{OH} = -6mA on: $\overline{CS}$5-0	2.4		V	
Output Low Voltage	V_{OL}	I_{OL} = 6mA on: $\overline{CS}$5-0		0.4	V	

7.1.4.10 Interrupt Controller

Parameter	Symbol	Condition	Min	Max	Unit	Notes
Output High Voltage	V_{OH}	I_{OH} = -12mA on: $\overline{INTA}$	2.4		V	
Output Low Voltage	V_{OL}	I_{OL} = 12mA on: $\overline{INTA}$		0.4	V	

7.1.4.11 3-Wire I/O (& Access.bus)

Parameter	Symbol	Condition	Min	Max	Unit	Notes
Output High Voltage	V_{OH}	I_{OH} = -12mA on: SO, SI, SCLK	2.4		V	
Output Low Voltage	V_{OL}	I_{OL} = 12mA on: SO, SI, SCLK		0.4	V	

7.2 General AC Specifications

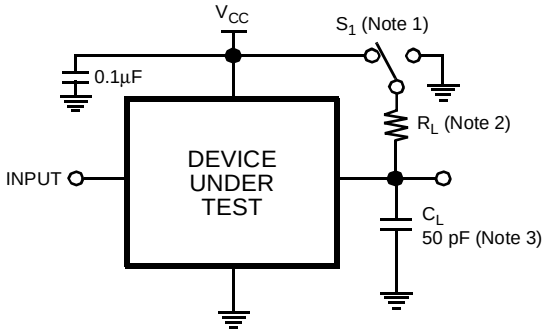
AC TEST CONDITIONS

Note 1: $S_1 = V_{CC}$ for t_{PZL}, and t_{PLZ} measurements
S_1 = GND for t_{PZH}, and t_{PHZ} measurements
S_1 = Open for push pull outputs

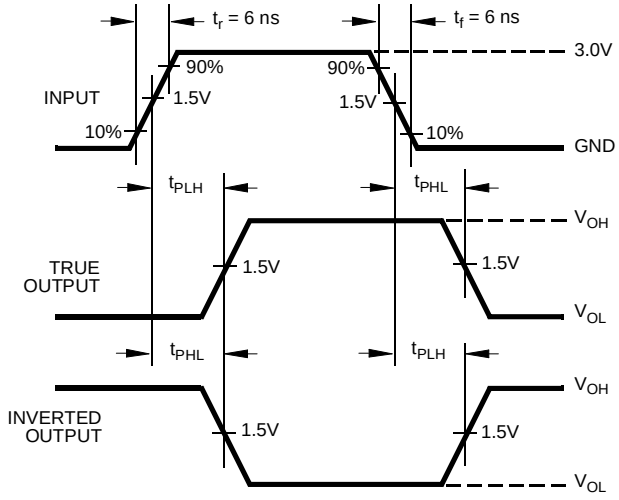
Note 2: R_L = 1.1k

Note 3: C_L includes scope and jig capacitance

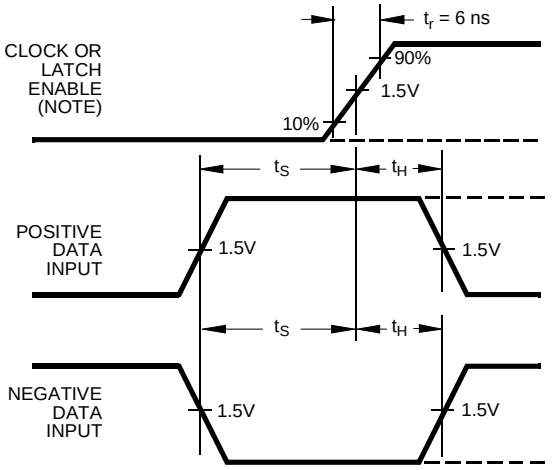
Test Circuit for Output Tests



Propagation Delay Waveforms

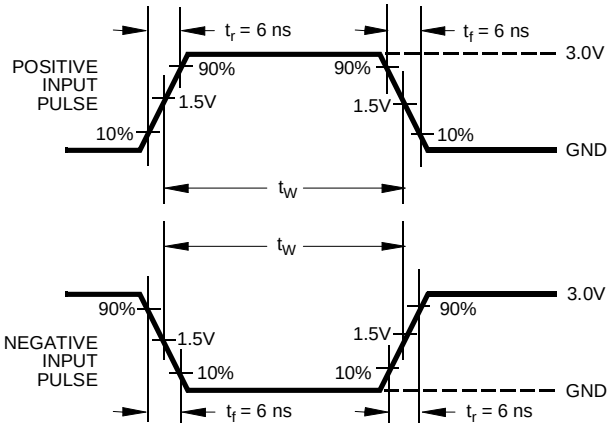


Setup and Hold Time Waveforms



Note: Waveform for negative edge sensitive circuits will be inverte

Input Pulse Width Waveforms Except for Clock Pins



TRI-STATE Output Enable and Disable Waveforms

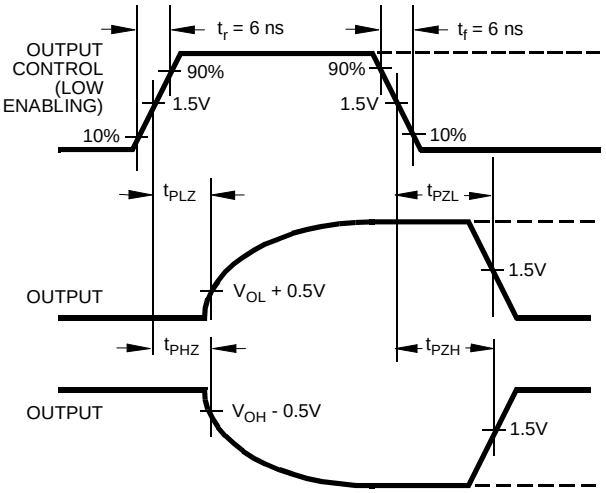
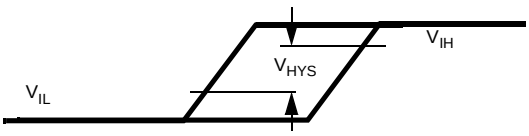


Figure 7-1 Switching Characteristic Measurement Waveforms



V_{HYS} = 200mV
Switching thresholds not specified

Figure 7-2 More Switching Specifications

7.2.1 Power Ramp Times

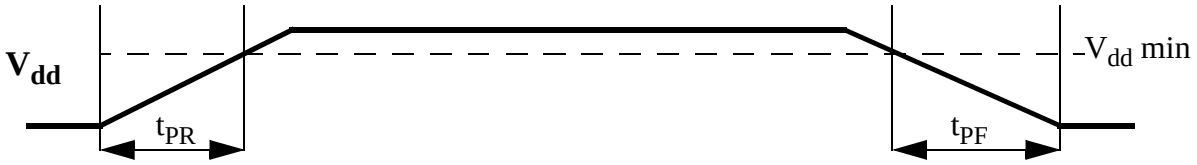


Figure 7-3 Power Supply Rise and Fall

Table 7-1: V_{dd} Rise and Fall Times

Symbol	Parameter	Min	Max	Unit
t_{PF}	V_{dd} falling time from 4.5V to 0 Volts	5		ms
t_{PR}	V_{dd} rising time from 0V to 4.5 Volts	5		ms

Note: The rising/falling rate is assumed linear.

7.2.2 PWRGOOD and Power Rampdown Timing

Table 7-2: V_{dd} Rampdown vs. PWRGOOD

Symbol	Parameter	Min	Max	Unit
t_{VPG}	V_{dd} (4.5V) to PWGOOD high	1		µs
t_{PGV}	PWGOOD falling to V_{dd} (4.5V)	1		µs

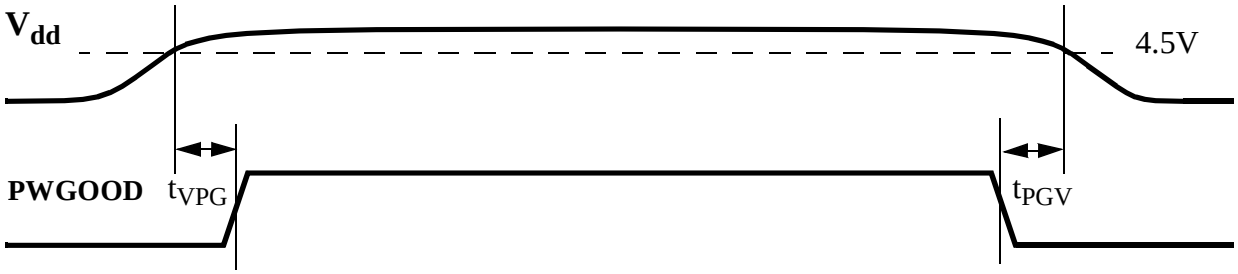


Figure 7-4 PWGOOD in relation to V_{dd}

Note: The rising/falling rate is assumed linear.

7.3 AC Switching Specifications

The following pages list some of the preliminary AC Specifications for the **NS486SXL**. All parameters are listed in alphabetical order according to their Symbol.

The Tables consist of the following:

Parameter - A short description of the specification being documented.

Symbol - A quick reference between the timing diagram and the Table entries.

Formula - An equation, which in addition to the Minimum and Maximum Specifications can be used to determine the actual timing provided at any operating frequency.

Min. - Minimum Specification when added to the value produced by the formula.

Max. - Maximum Specification when added to the value produced by the formula.

How to calculate the actual specification at a given frequency:

In the formula column, one will see many formulae, which contain the variable T. The T represents one period (or one T-state) of the CPU Clock. So if the CPU is running at 25 MHz, T is equivalent to 40 nsec; similarly if the CPU is running at 20 MHz, T is equivalent to 50 nsec.

EXAMPLE: Calculate the minimum guaranteed Column Address Setup Time

At 25 MHz:	Formula + Min. Spec.	=
	(0.5T) + (-20 nsec)	=
	0.5(40 nsec) + (-20 nsec)	=
	20 nsec - 20 nsec	= 0 nsec
At 20 MHz:	Formula + Min. Spec.	=
	(0.5T) + (-20 nsec)	=
	0.5(50 nsec) + (-20 nsec)	=
	25 nsec - 20 nsec	= 5 nsec

As the frequency varies, so will many of the specifications. One should always calculate the specification based on the CPU's operating frequency.

7.3.1 DRAM Interface Timing Specification.

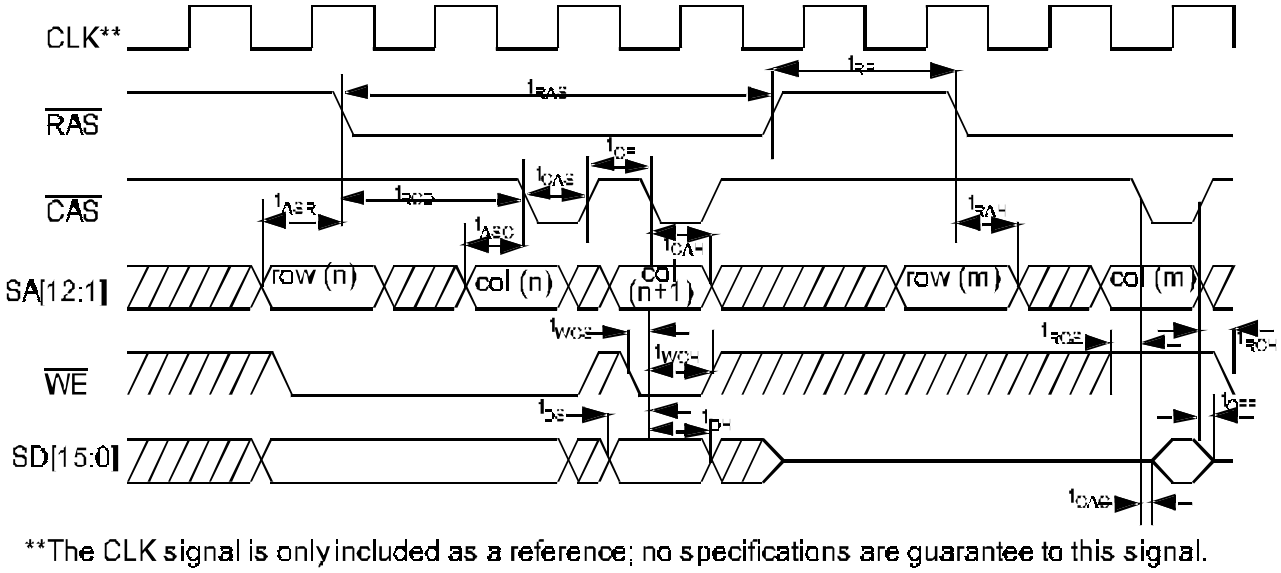


Figure 7-5 DRAM Timing Diagram

Table 7-3: 4 Cycle Page Miss Preliminary Specifications

Parameter	Symbol	Formula	Min	Max
Column Address Setup Time	t_{ASC}	0.5T +	-20	
Row Address Setup Time	t_{ASR}	0.5T +	-20	
Access Time From CAS	t_{CAC}	0.5T +		-5
Column Address Hold Time	t_{CAH}	0.5T +	-5	
$\overline{CAS}$ Pulse Width	t_{CAS}	0.5T +	0	10
Page Mode $\overline{CAS}$ Precharge	t_{CP}	0.5T +	-10	
Write Data Hold Time	t_{DH}	0.5T +	-5	
Write Data Setup Time	t_{DS}	0.5T +	-20	
Read Data Valid Hold Time	t_{OFF}		0	
$\overline{RAS}$ Pulse Width	t_{RAS}	2.5T +	-15	Progr'm'ble
Row Address Hold Time	t_{RAH}	0.5T +	-10	
$\overline{RAS}$ to $\overline{CAS}$ Delay Time	t_{RCD}	1.5T +	-20	

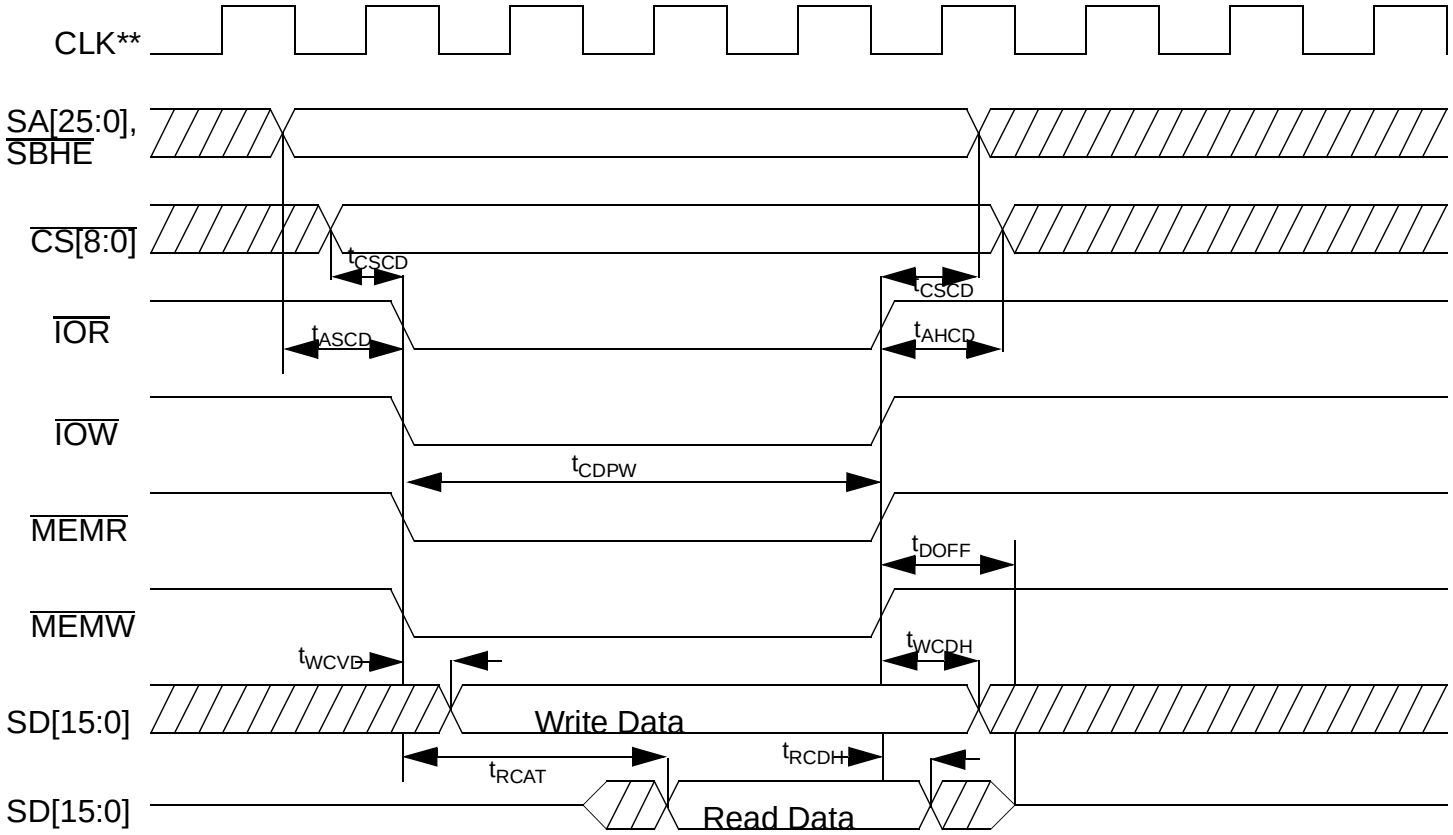
Table 7-3: 4 Cycle Page Miss Preliminary Specifications

Parameter	Symbol	Formula	Min	Max
Read Command Hold Time	t_{RCH}		0	
Read Command Setup Time	t_{RCS}	0.5T +	-20	
$\overline{RAS}$ Precharge Time	t_{RP}	1.5T +	-10	
Write Command Hold TIme	t_{WCH}	0.5T +	-5	
Write Command Setup Time	t_{WCS}	0.5T +	-20	

Table 7-4: 3 Cycle Miss Preliminary Specifications

Parameter	Symbol	Formula	Min	Max
Column Address Setup Time	t_{ASC}	0.5T +	-20	
Row Address Setup Time	t_{ASR}	0.5T +	-20	
Access Time From $\overline{CAS}$	t_{CAC}	0.5T +		-5
Column Address Hold Time	t_{CAH}	0.5T +	-5	
$\overline{CAS}$ Pulse Width	t_{CAS}	0.5T +	0	10
Page Mode $\overline{CAS}$ Precharge	t_{CP}	0.5T +	-10	
Write Data Hold Time	t_{DH}	0.5T +	-5	
Write Data Setup Time	t_{DS}	0.5T +	-20	
Read Data Valid Hold Time	t_{OFF}		0	
$\overline{RAS}$ Pulse Width	t_{RAS}	2.0T +	-15	PROG
Row Address Hold Time	t_{RAH}	0.5T +	-10	
$\overline{RAS}$ to $\overline{CAS}$ Delay Time	t_{RCD}	1.0T +	-20	
Read Command Hold Time	t_{RCH}		0	
Read Command Setup Time	t_{RCS}	0.5T +	-20	
$\overline{RAS}$ Precharge Time	t_{RP}	1.0T +	0	
Write Command Hold TIme	t_{WCH}	0.5T +	-5	
Write Command Setup Time	t_{WCS}	0.5T +	-20	

7.3.2 ISA-like Bus Cycles Timing Specification



**The CLK signal is only included as a reference; no specifications are guarantee to this signal.

Figure 7-6 ISA-like Bus Timing Diagram

Table 7-5: No Command Delay ISA-like Bus Specifications

Parameter	Symbol	Formula	Min	Max
Address Hold Time from $\overline{CMD}$	t_{AHCD}	1.0T +	-20	
Address Setup TIme to $\overline{CMD}$	t_{ASCD}	1.0T +	-20	
Command Pulse Width	t_{CDPW}	1.0T + (Wait)T +	-10	
Chip Select Hold Time from $\overline{CMD}$	t_{CHCD}	1.0T +	-25	
Chip Select Setup Time to $\overline{CMD}$	t_{CSCD}	1.0T +	-40	
Read Data TRI-STATE	t_{DOFF}	1.0T +		-25
Read $\overline{CMD}$ Data Access TIme	t_{RCAT}	1.0T + (Wait)T +		-30
Read $\overline{CMD}$ Data Hold TIme	t_{RCDH}		0	

Table 7-5: No Command Delay ISA-like Bus Specifications

Parameter	Symbol	Formula	Min	Max
Write $\overline{CMD}$ Data Hold Time	t_{WCDH}	1.0T +	-25	
Write $\overline{CMD}$ to Valid Data	t_{WCVD}			5
Write Command Setup Time	t_{WCS}	0.5T +	-20	

NOTE: The value of (Wait) in the above formulae, is the number of programmed wait states associated with that access cycle (default value is 7, but may be programmed to 0-7).

Table 7-6: One Programmed Command Delay ISA-like Bus Specifications

Parameter	Symbol	Formula	Min	Max
Address Hold Time from $\overline{CMD}$	t_{AHCD}	1.0T +	-20	
Address Setup TIme to $\overline{CMD}$	t_{ASCD}	2.0T +	-20	
Command Pulse Width	t_{CDPW}	1.0T + (Wait)T +	-10	
Chip Select Hold Time from $\overline{CMD}$	t_{CHCD}	1.0T +	-25	
Chip Select Setup Time to $\overline{CMD}$	t_{CSCD}	2.0T +	-40	
Read Data TRI-STATE	t_{DOFF}	1.0T +		-25
Read $\overline{CMD}$ Data Access TIme	t_{RCAT}	1.0T + (Wait)T +		-30
Read $\overline{CMD}$ Data Hold TIme	t_{RCDH}		0	
Write $\overline{CMD}$ Data Hold Time	t_{WCDH}	1.0T +	-25	
Write Valid Data to CMD[NOTE_2]	t_{WCVD}	1.0T +	-5	
Write Command Setup Time	t_{WCS}	0.5T +	-20	

NOTE: The value of (Wait) in the above formulae, is the number of programmed wait states associated with that access cycle (default value is 7, but may be programmed to 0-7).
NOTE_2: For this case Valid Write Data Sets-up to the leading edge of the Command Strobe.

7.3.3 Ready Feedback Timing Specifications

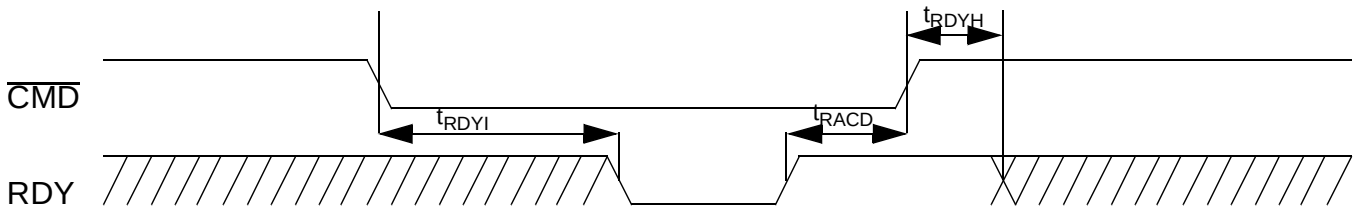


Figure 7-7 Ready Feedback Timing Diagram

Table 7-7: Ready Signal Timing Specifications

Parameter	Symbol	Formula	Min	Max
RDY Active to $\overline{CMD}$ rising	t_{RACD}	(E_RDY)T +	0	
RDY Hold Time from $\overline{CMD}$	t_{RDYH}		0	
$\overline{CMD}$ to RDY Inactive Feedback	t_{RDYI}	1.0T + (Wait)T +		-30

NOTE: The value of (Wait) in the above formulae, is the number of programmed wait states associated with that access cycle (default value is 7, but may be programmed to 0-7). The value of (E_RDY) in the above formulae, is the number of programmed extended ready states associated with every access cycle (default number is 2, but may be programmed to 0-2)

7.3.4 OSCX1 AC Specification

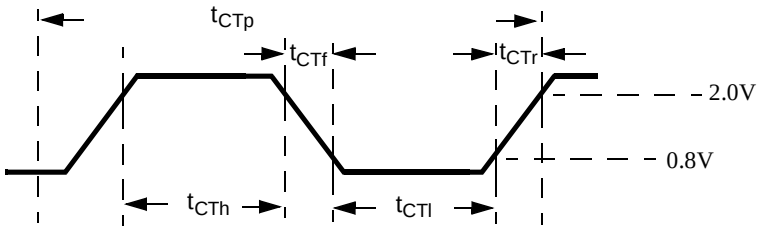


Figure 7-8 TTL Clock Input Timing Diagram

Table 7-8: TTL Clock Input Specification

Symbol	Description	Min	Max	Unit
t_{CTp}	CTTL Clock period	40	870	ns
t_{CTh}	CTTL high time (Note)	(0.5 X t_{CTp}) - 4		ns
t_{CTl}	CTTL low time (Note)	(0.5 X t_{CTp}) - 4		ns
t_{CTr}	CTTL rise time		4	ns
t_{CTf}	CTTL fall time		4	ns

Note: Except for the cycle in which the core frequency is changed. In this cycle, t_{CTh} and t_{CTl} relate to different t_{CTp} cycles.

7.3.4.1 PIC AC Specs

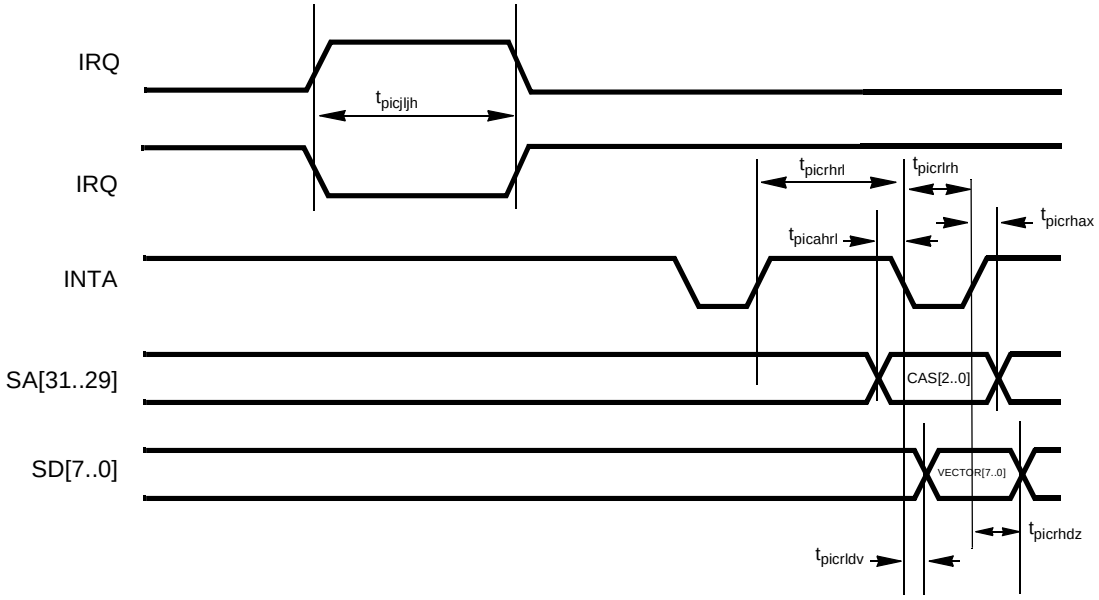


Figure 7-9 PIC Timing Diagram

Table 7-9: PIC Timing Specifications

Symbol	Parameter	Min	Typ	Max
$t_{picjljh}$		100		
$t_{picahrl}$		0		
$t_{picrlrh}$		235		
$t_{picrhax}$		0		
$t_{picrldv}$				200
$t_{picrhdz}$		10		
$t_{picrhrl}$		100		

7.3.4.2 MICROWIRE (3-Wire) & Access.bus

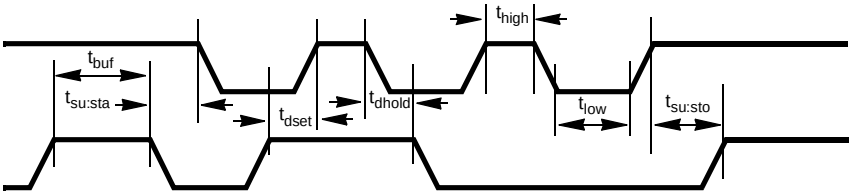


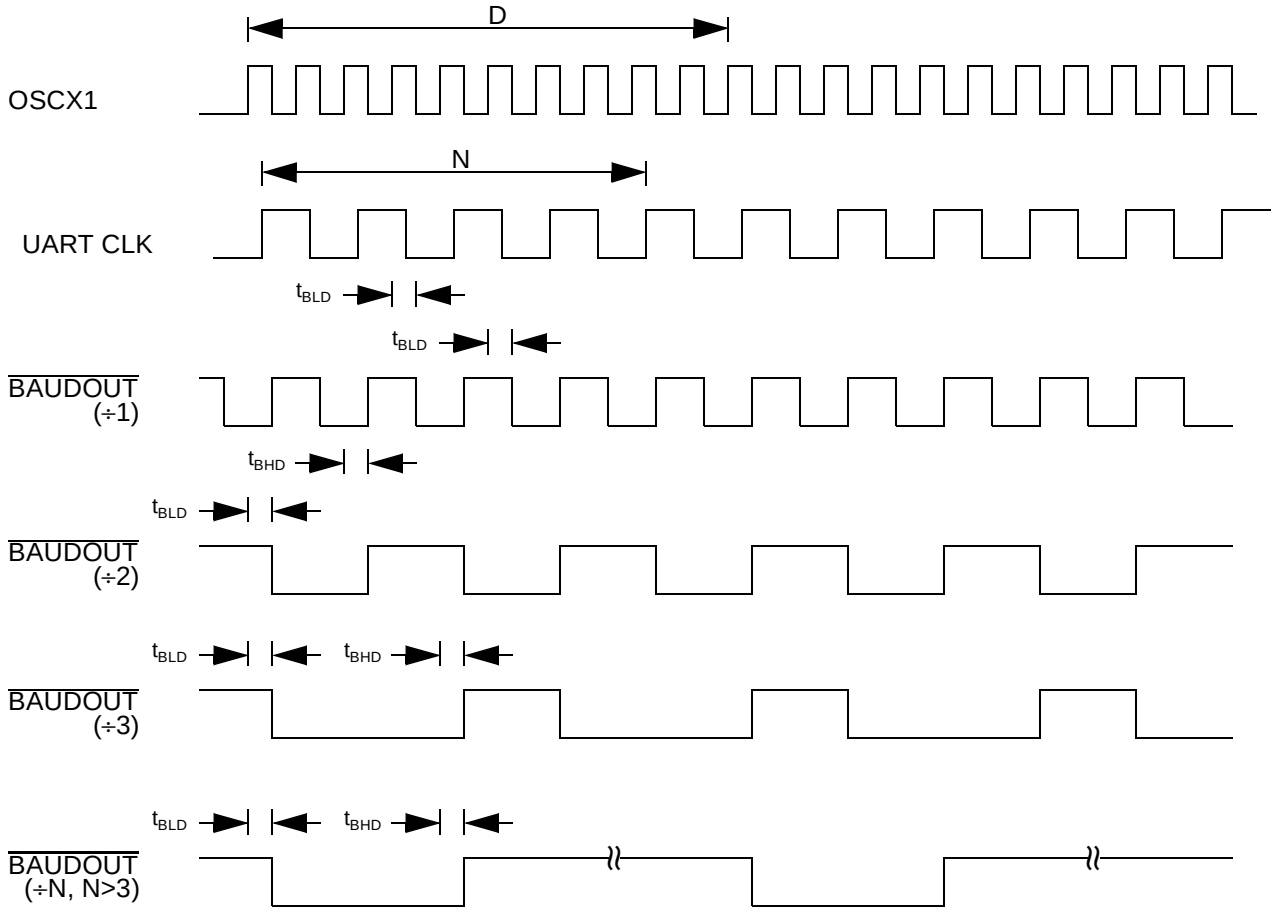
Figure 7-10 Access.bus Timing Diagram

Table 7-10: Access.Bus Timing Specifications

Symbol	Parameter	Formula	Min	Max
f_{sclk}	SCLK clock frequency			100 KHz
t_{buf}	Bus free time between STOP and START condition		4.7 us	
t_{low}	Low period of the SCLK clock		4.7 us	
t_{high}	High period of the SCLK clock		4.0 us	
t_{dhold}	Data hold time		250	
t_{dset}	Data setup time		250	
$t_{su:sto}$	Setup time for STOP condition		4.0 us	
$t_{su:sta}$	Hold time for START condition		4.7 us	

7.3.4.3 FIFO UART

Symbol	Parameter	Conditions	Min	Max	Units
D	Osc Clock Divider		1	63	Clks
N	Baud Divisor		1	65535	Clks
t_{BHD}	Baud Output Positive Edge Delay			56	ns
t_{BLD}	Baud Output Negative Edge Delay			56	ns



Symbol	Parameter	Conditions	Min	Max	Units
t_{IRTXW}	IRTX Pulse Width		1.6 µs	3/16	BAUD OUT Cycles
t_{IRRXW}	IRRX Pulse Width		1.6 µs	6/16	BAUD OUT Cycles

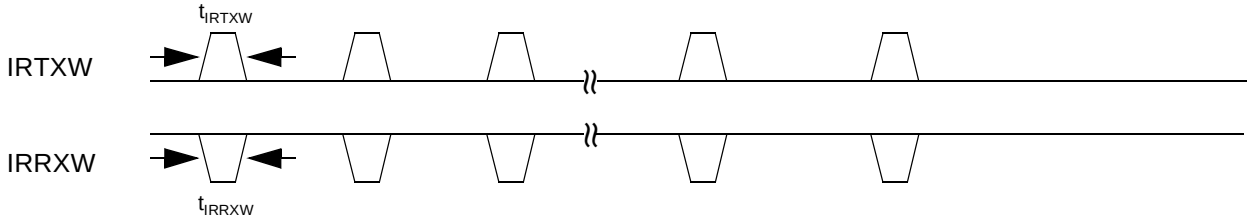


Figure 7-11 UART Baud Rate and Infrared Clocks

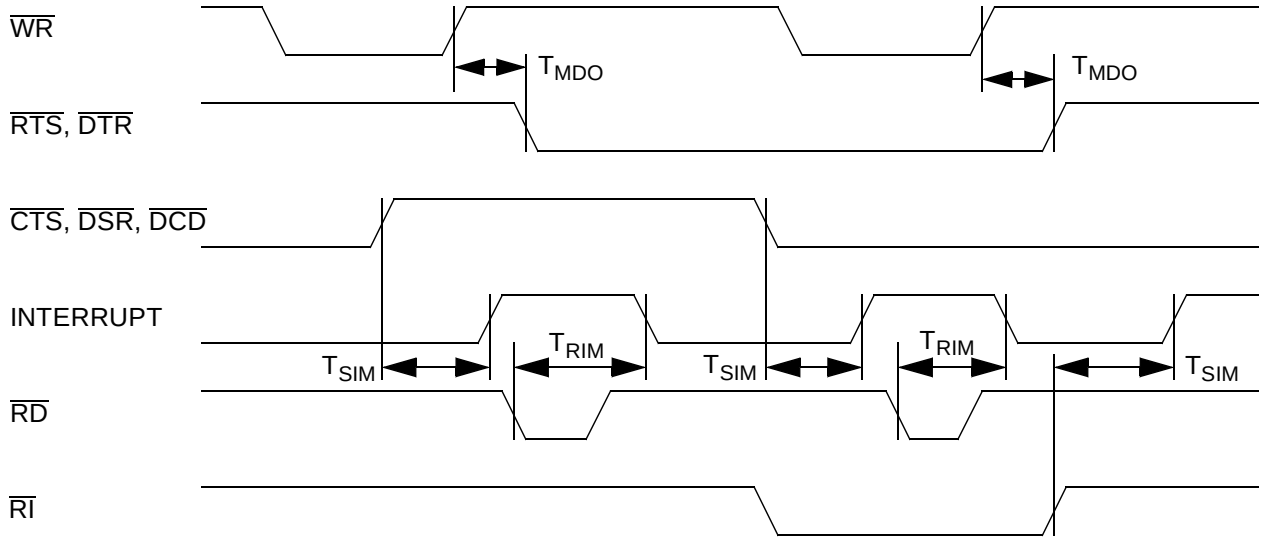


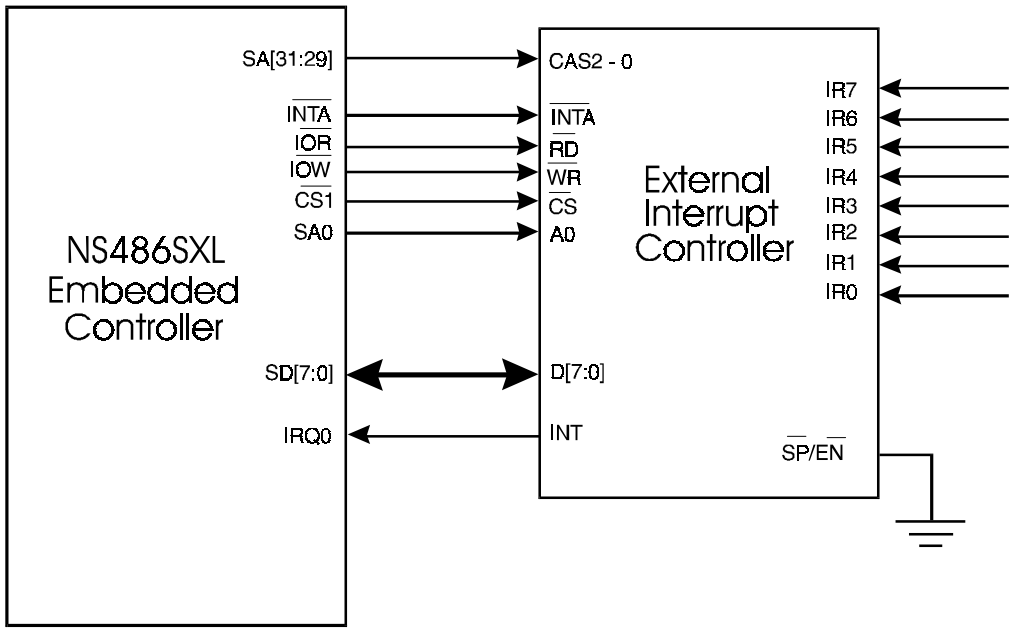
Figure 7-12 UART MODEM Control Timing

Deliberately Blank Page

Appendix A: External Interrupt Controller

Connecting a cascaded external interrupt controller to NS486SXL is shown in the following diagram.

Figure A-1 External Interrupt Controller Connection



Note: This diagram shows one method to connect a cascaded external interrupt controller up to the NS486SXL. However, any of the IRQ[n] pins can be used (as long as that pin is multiplexed to the NS486SXL's internal master interrupt controller) and any one of $\overline{CS8}$-$\overline{CS1}$ signals. IRQ0 and $\overline{CS1}$ are provided as examples only.

For a system that uses data buffering, the buffers will obviously need to be enabled for reading during an interrupt acknowledge cycle for an external slave. One way of achieving this is to put the external interrupt controller into slave mode through software, and use the BDIR signal to enable the system data read buffers.

This Page Intentionally Left Blank.

Appendix B: Dealing with Unused Signals on the NS486SXL

This document answers the common question about what to do with unused signals on the ‘SXL.

The basic rule followed is: Any signal that is an input should be pulled to its inactive state (or Pulled high if it has no effect if it is not enabled). This is necessary so the input signal does not float to the input threshold (turning on the input's p- and n-channels and getting current flow through both channels). Often this is termed as input leakage current, but it truly is current lost from input flow through switching current.

Please note that recommendations are made in this document based on the power up state of a given signal. Some of the treatments may seem to contradict the functional description of the pin in the pinout description section of the databook. In general those are signals that power-up in a TRI-STATE condition, even though they are declared as outputs in the Databook. In their enabled functionality, they will be outputs, but only after being enabled.

At a minimum, the following signals will usually be used in the system:

- SA[31:0]
- SD[15:0]
- $\overline{\text{ALS}}$
- $\overline{\text{SBHE}}$
- $\overline{\text{IOR}}$
- $\overline{\text{IOW}}$
- $\overline{\text{MEMR}}$
- $\overline{\text{MEMW}}$
- $\overline{\text{CS16}}$
- RDY
- $\overline{\text{RAS[1:0]}}$
- $\overline{\text{CASH[1:0]}}$
- $\overline{\text{CASL[1:0]}}$
- $\overline{\text{WE}}$
- Vdd
- Vss
- PWGOOD
- OSCX1 (and OSCX2 if a crystal network is used, but no OSCX2 if OSCX1 is driven by a TTL-oscillator)
- $\overline{\text{CS[0]}}$

No Pullup or Pulldown resistors should be connected to the following signals:

- $\overline{\text{RAS[1:0]}}$
- $\overline{\text{CASH[1:0]}}$
- $\overline{\text{CASL[1:0]}}$
- $\overline{\text{WE}}$
- Vdd
- Vss
- OSCX1
- $\overline{\text{CS[0]}}$

Cascade Master Mode

If Cascade Master Mode is used and an External Master takes control of the ISA-like Bus, then the following signals should have Pullups on them:

- SA[31:18] and SA[16:0]
- $\overline{\text{IOR}}$
- $\overline{\text{IOW}}$
- $\overline{\text{MEMR}}$
- $\overline{\text{MEMW}}$

Otherwise, these signals do not need Pullup or Pulldown resistors.

Data Bus

The ISA-like data bus (SD[15:0]) should be pulled up with 10k resistors so that it does not float during long periods of inactivity. SD[15:0] defaults to be an input into the NS486SXL and thus will float (if it is not pulled up) during IDLE times.

Address Bus

- **SA[17] - To ensure correct operation of the NS486SXL, the SA[17] pin *must* be pulled high with a 10K Ohm resistor. If this is not done and the pin is left either floating or is pulled low, unreliable and undefined operation will occur.**

Strapping Options

- $\overline{\text{SBHE}}$ - Sampled at the rising edge of PWGOOD to determine Boot ROM interface size (8-bit if

pulled down, 16-bit if pulled up). Choose appropriately.

- SA[16] - Defines the positin of the Rom BIOS with respect to the 74x245 buffers. This pin should be either tied high or low with a 10K Ohm Resistor as per the following definition:
  Low : Indicates to the NS486SXL that the ROM BIOS is remote, ie. on the far side of the buffers with respect to the 'SXL.
  High : Indicates to the NS486SXL that the ROM BIOS is local, ie. on the near side of the buffers with respect to the 'SXL.

Pulled High

If the following signals are unused, pulling them high will prevent them from floating to input threshold and resulting in high input leakage.

- $\overline{\text{CS16}}$ - Should always be pulled high.
- RDY - Should always be pulled high.
- $\overline{\text{TEST}}$ - Should always be pulled high.
- $\overline{\text{EACK}}$ - Should always be pulled high.
- $\overline{\text{IRQ[7:0]}}$ - Should always be pulled high.

Pulled Low

It is recommended that the following signal be pulled low with a 10K Ohm resistor.

- HOLD

Tie Low If Unused

- NMI
- Vbat
- RTCX1 (RTCX2 should be a No Connect if the RTC is unused and RTCX1 is tied low.)

Pull High If Unused

- DPH and DPL - These signals will be TRI-STATE if DRAM Parity is not enabled. So to prevent these signals from floating to input threshold and resulting in high input leakage, it is suggested that these signals are pulled high if DRAM Parity is not to be enabled.
- Rx - This will be an input and if not pulled high it may float to input threshold and result in high input leakage.
- T0 - This pin defaults to being an input and pulling it high will prevent it from floating to input threshold and resulting in high input leakage (It may also be configured as an output via Boot software to prevent this situation, if so desired.)
- T1 - This pin defaults to being an input and pulling it high will prevent it from floating to input threshold and resulting in high input leakage (It may also be configured as an output via Boot software to prevent this situation, if so desired.)
- SI - should be pulled high to prevent the signal from floating to input threshold and resulting in high input leakage when not being used.

No Connection When Unused

- UCLK - This pin defaults to being an output (oscillating) upon reset, so it can be left unconnected.
- RESET
- $\overline{\text{RESET}}$
- SYSCLK
- $\overline{\text{EREQ}}$
- $\overline{\text{DRV}}$
- $\overline{\text{INTA}}$
- Tx
- $\overline{\text{CS[8:1]}}$
- $\overline{\text{HLDA}}$
- $\overline{\text{MAE}}$
- $\overline{\text{D/C}}$